DeepDecipher: Accessing and Investigating Neuron Activation in Large Language Models

Albert Garde[*]
Apart Research
albertsgarde@gmail.com

Esben Kran[*]
Apart Research
esben@apartreserach.com

Fazl Barez
Apart Research
Edinburgh Centre for Robotics
Department of Engineering Sciences, University of Oxford
fazl@robots.ox.ac.uk

Abstract

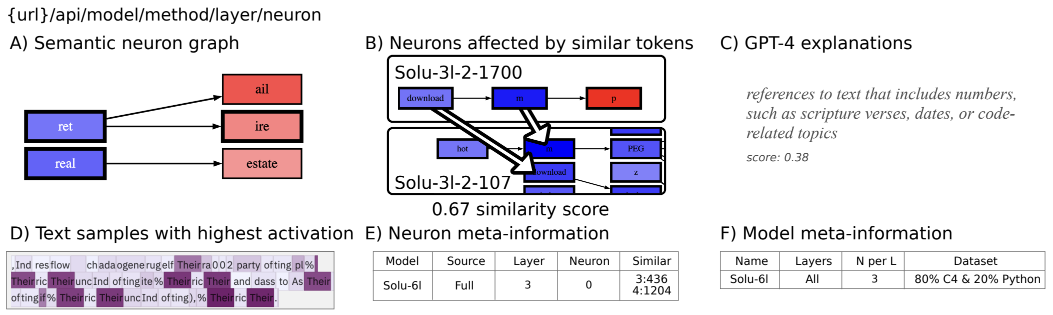
As large language models (LLMs) become more capable, there is an urgent need for interpretable and transparent tools. Current methods are difficult to implement, and accessible tools to analyze model internals are lacking. To bridge this gap, we present DeepDecipher - an API and interface for probing neurons in transformer models' MLP layers. DeepDecipher makes the outputs of advanced interpretability techniques for LLMs readily available. The easy-to-use interface also makes inspecting these complex models more intuitive. This paper outlines DeepDecipher's design and capabilities. We demonstrate how to analyze neurons, compare models, and gain insights into model behavior. For example, we contrast DeepDecipher's functionality with similar tools like Neuroscope and OpenAI's Neuron Explainer. DeepDecipher enables efficient, scalable analysis of LLMs. By granting access to state-of-the-art interpretability methods, DeepDecipher makes LLMs more transparent, trustworthy, and safe. Researchers, engineers, and developers can quickly diagnose issues, audit systems, and advance the field.

1 Introduction

Explainability and safety in AI are becoming increasingly important as language models like GPT-4 make up the next generation of software (Baktash and Dawodi, 2023; Wu et al., 2023). Methods in explainable AI, making AI more understandable for human users, often draw inspiration from interpretability research. Recent work has shifted attention from vision models (Erhan et al., 2009; Simonyan et al., 2014) to Transformer and generative models (Vaswani et al., 2023; Elhage et al., 2021).

Transformers contain attention and multi-layer perceptron (MLP) layers (Brown et al., 2020). The attention layers (that "transfer information" between sections of the text in latent space) have received more attention within the interpretability research community (Olsson et al., 2022), while the MLP layers (that non-linearly modify this information) remain underexplored (Elhage et al., 2022a). Given their role in complex language tasks, the lack of targeted interpretability tools for MLPs is a glaring gap in ensuring explainable AI becomes the norm. In this paper, we introduce and describe DeepDecipher, a tool to visualize, interpret, and explain MLP neuron activation

[*]Equal contribution.


{url}/api/model/method/layer/neuron
A) Semantic neuron graph
ret
real
ail
ire
estate
B) Neurons affected by similar tokens
Solu-3l-2-1700
download
m
p
Solu-3l-2-107
hot
PEG
z
0.67 similarity score
C) GPT-4 explanations
D) Text samples with highest activation
E) Neuron meta-information
F) Model meta-information

references to text that includes numbers, such as scripture verses, dates, or code-related topics

score: 0.38

,Ind res flow chadaogenerugelf Their ra 0 0 2 party of ting pl % Their ric Their unc Ind of ting ite % Their ric Their and dass to As Their of ting if % Their ric Their unc Ind of ting), % Their ric Their.

Model	Source	Layer	Neuron	Similar
Solu-6l	Full	3	0	3:436 4:1204

Name	Layers	N per L	Dataset
Solu-6l	All	3	80% C4 & 20% Python

Figure 1: Data sources available in the DeepDecipher user interface and API: **A)** Graphs encapsulating which tokens influence neuron activation (Foote et al., 2023). **B)** The graphs from (A) are compared, and co-occurrence of token nodes is used to calculate the similarity score as the two-way maximum proportion of overlapping nodes. **C)** GPT-4-based automated descriptions of the semantics associated with neuron activation (Bills et al., 2023). **D)** A list of snippets that this neuron activates highly to. Each snippet consists of 1024 tokens colored according to the neuron activation on that token (Nanda, 2022a). **E)** Statistics and meta-information about the neuron **F)** Meta-information about the model and layers

in large language models. It eases access to explanations of low-level causal components of open language models (Radford et al., 2019; Biderman et al., 2023) based on the principles of mechanistic interpretability (Olah et al., 2020). DeepDecipher does not introduce novel methods for neuron explanations but makes recent interpretability research results available in an accessible user interface and API.

2 Functionality

DeepDecipher aims to simplify access to the data generated by existing interpretability methods, specifically methods that provide information on the behaviour of MLP neurons in transformer-based language models. It does this both through an API and a visual interface that allows users to understand when and why an MLP neuron activates. DeepDecipher provides access to the following modern MLP interpretability methods.

Neuron to Graph (N2G) is a method to create a graph of tokens sequences that affect a specific MLP neuron's activation. It is generated by backtracking from the most activating tokens and finding the tokens that affect the activation on the final token the most by taking examples of sequences that lead to activation and replacing tokens until one finds 1) the tokens most important for activation on 2) the end tokens where the neuron activates.

Neuroscope is a database of the 20 1024 token sequences that each neuron of a model activates the most to. The 1024 token length sequences are sampled from OpenWebText (Gokaslan and Cohen, 2019), the Pile (Gao et al., 2020), C4 and Python depending on the training dataset of the models. The model activations on runs over the token sequences are used to generate this dataset: For OpenWebText, each model is run over 9b tokens; for the Pile, each model is run over 2b tokens and for C4 and Python, each model is run over 1.4b tokens of C4 and 0.3b tokens of Python.

Neuron Explainer is an automated interpretability method that uses GPT-4 to explain which categories of token sequences a neuron responds to (e.g. "references to movies, characters, and entertainment.") and evaluates how well GPT-4 can predict that neuron's activation based on this description. They apply their method to the open GPT-2 XL model.

Based on the research behind N2G, we also introduce a **search function** to find the neurons that respond the most to a particular token along with a **neuron similarity** based on the same N2G data. The search queries the N2G graphs for the token and similarity is scored based on co-occurring tokens in the N2G graphs of two neurons.

Website: The website provides an interactive visual interface to enable rapid hypothesis testing and model understanding. As shown in Figure 1, visual representations are provided for all model data accessible through the API. This allows users to visually explore the inner workings of the model.

Specifically, it is possible to navigate through the model by neuron index and search for neurons that activate on specific tokens or concepts. The visualizations make it easy to identify patterns, trends, and relationships in the model's representations and processing. Users can quickly validate assumptions and gain insights by observing model behavior. The interface provides access to different types of data about the model, as outlined in Table 2. With these visualization and data analysis tools, researchers can interactively probe the model to test hypotheses about its functioning. The hands-on exploration facilitates rapid intuition development about model mechanisms.

API: The data found on the website is all available in JSON format through the API. Endpoints take the form `/api/<model>/<service>/<layer>/<neuron>` where `<service>` is the type of data e.g. `neuron2graph`, `neuroscope` or `all`.

For examples, see Section C or the repository[*]. **Extendable framework:** Everything is designed to be easily extendable with additional models and interpretability methods. The server is designed with scalability in mind to handle many simultaneous users.

	DeepDecipher	Neuroscope	Neuron Explainer
Unique Contribution	Search and availability	Activating dataset examples	GPT-4 neuron explanations
API Output	JSON	XML (HTML)	JSON
Speed (requests/second)	56 (API)*	87 (web page)	53 (API)
Data Available	Neuron to Graph, Neuroscope, Model Explanations (extendable)	Neuroscope	Model Explanations
Models Available	25 (extendable)	25	2

Table 1: Comparison of DeepDecipher, Neuroscope, and OpenAI Neuron Explanations. Note that all measurements are mainly bottle-necked by client download speed and do therefore not represent the capacity of the server. *The stated API speed is the lowest speed across services. See table 2 for details.

3 Available data

We currently support 3 services and 25 models. The models supported are the same as those supported by Neuroscope (Nanda, 2022a), which include a range of small `solu` and `gelu` models (Nanda et al., 2023), 4 GPT-2 models (Radford et al., 2019), and 3 `pythia` models (Biderman et al., 2023). Full details are available on the website[*]. Neuroscope is supported for all models, Neuron2Graph is on its way for all models, and Neuron Explainer is only available for GPT-2 Small and GPT-2 XL.

4 Usage

DeepDecipher allows users with a comprehensive range of capabilities, including:

1. DeepDecipher offers API access to state-of-the-art data concerning the functionality, activation patterns, and activation dependencies of individual MLP neurons in various open large language models. Additionally, it provides the flexibility to easily expand this API to accommodate new models.

[*]https://github.com/apartresearch/deepdecipher

[*]`https://deepdecipher.org/viz`

	Description	Models	API Speed
Neuron to Graph	Graph illustrating what tokens activate or modulate a neuron (Foote et al., 2023)	`solu-6l-pile` (all coming*)	388
Neuroscope	20 top-activating 1024-token sequences based on the training data (Nanda, 2022a)	All	56
Neuron Explainer	Explanation by GPT-4 and score of the explanation (Bills et al., 2023)	`gpt2-small` and `gpt2-xl`	384

Table 2: Data available in the model. API speed is measured as requests per second. Note that all measurements are mainly bottle-necked by client download speed and do therefore not represent the capacity of the server. *Neuron to Graph data is currently only available for a single model, but we plan to extend this to all models very soon.

2. With DeepDecipher, you can efficiently search for neurons that respond to specific tokens within these models.
3. DeepDecipher creates dedicated web pages for each model, layer, and MLP neuron, complete with static links that grant users convenient access to the API data. Furthermore, DeepDecipher can be deployed locally, making it compatible with proprietary models.

DeepDecipher's functionality proves useful under many scenarios (listed below). For mechanistic interpretability researchers, the API provides access to useful data on MLP neuron functionality without the need to re-run the computationally expensive processing (e.g., $O(n^{2/3})$ for (Bills et al., 2023)). The upcoming EU AI Act (European Commission, 2021) ushers in an era of increased transparency for "high-risk AI" models. When these models are deployed in sensitive applications, users will have the right to understand why certain outputs or decisions were made. As engineers prepare for this changing regulatory environment, tools like DeepDecipher can provide helpful capabilities to enable model interpretability. Web pages, API, and search functionality provide the scaffolding to explain the output of neural networks (Montavon et al., 2018; Conmy et al., 2023). As an engineer and researcher, we can rapidly test theories about language model comprehension. Below, we present some case studies. Using the search functionality for the `solu-6l-pile` model*:

- Seeing how Spanish is represented compared to English, finding that no tokens encode for `hola` (0) compared to `hello` (7) which corresponds to the training dataset being filtered for the English language (Gao et al., 2020).
- Evaluating the emotional specificity of a model proxied by how many neurons respond to happy (9), sad (5), hate (3), love (20), morose (0) and other emotions, corroborating benchmarks for emotional intelligence of large language models (Wang et al., 2023).
- Understanding how models differentiate between homographs, e.g., `Apple` the company and `apple` the fruit. By looking at the token activation dependency graphs on each page, we can see that of the 8 neurons responding to `apple`, 5 are about the company, 2 are about the fruit and 1 is ambiguous, showing disambiguation between the two concepts (Mikolov et al., 2013).

For application developers using language models, providing easy access for users to embedded and linked explanations for the neurons that are involved in actions the AI system takes enables a more secure and explainable user-AI interface (Vig, 2019). See Appendix C for documentation on using the API.

5 Discussion and future work

The DeepDecipher project exposes the internals of neural networks in an interface and aims to be a robust and reliable application for users interested in understanding what happens within

*Available on the website's `https://deepdecipher.org/viz/solu-6l-pile/all` solu-6l DeepDecipher page

LLMs. The vision for DeepDecipher is to be an integrated component to make any frontier AI system more explainable. This includes redesigning parts of the user interface, ensuring support for proprietary models, implementing better search functionality and partnering with researchers to provide a comprehensive database of models.

6 Conclusions

DeepDecipher provides a user-friendly way to understand how neurons in Transformer and MLP layers work. It aims to make complex AI systems easier to explain and more secure. Looking ahead, DeepDecipher is focusing on making it simpler to incorporate its features into other research projects and applications, becoming a mainstay of explainable applications.

Ackowledgements

We are grateful to Alex Foote for his continued assistance in getting Neuron2Graph integrated well in DeepDecipher. We thank Mikołai Kniejski for our long discussions on how to design the websites UX. Bart Buss, for his patience answering our questions on what features would be useful.

References

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. 2016. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*.

Omer Antverg and Yonatan Belinkov. 2022. On the Pitfalls of Analyzing Individual Neurons in Language Models. ArXiv:2110.07483 [cs].

Jawid Ahmad Baktash and Mursal Dawodi. 2023. Gpt-4: A review on advancements and opportunities in natural language processing.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, Usvsn Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. 2023. Pythia: A suite for analyzing large language models across training and scaling. *arXiv:2304.01373*.

Steven Bills, Nick Cammarata, Dan Mossing, Henk Tillman, Leo Gao, Gabriel Goh, Ilya Sutskever, Jan Leike, Jeff Wu, and William Saunders. 2023. Language models can explain neurons in language models.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *ArXiv*, abs/2005.14165.

Arthur Conmy, Augustine N. Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. 2023. Towards Automated Circuit Discovery for Mechanistic Interpretability. ArXiv:2304.14997 [cs].

Nadir Durrani, Hassan Sajjad, Fahim Dalvi, and Yonatan Belinkov. 2020. Analyzing Individual Neurons in Pre-trained Language Models. ArXiv:2010.02695 [cs].

Nelson Elhage, Tristan Hume, Catherine Olsson, Neel Nanda, Tom Henighan, Scott Johnston, Sheer ElShowk, Nicholas Joseph, Nova DasSarma, Ben Mann, Danny Hernandez, Amanda Askell, Kamal Ndousse, And Jones, Dawn Drain, Anna Chen, Yuntao Bai, Deep Ganguli, Liane Lovitt, Zac Hatfield-Dodds, Jackson Kernion, Tom Conerly, Shauna Kravec, Stanislav Fort, Saurav Kadavath, Josh Jacobson, Eli Tran-Johnson, Jared Kaplan, Jack Clark, Tom Brown, Sam McCandlish, Dario Amodei, and Christopher Olah. 2022a. Softmax linear units. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2022/solu/index.html.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. 2022b. Toy models of superposition.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2021/framework/index.html.

Dumitru Erhan, Y. Bengio, Aaron Courville, and Pascal Vincent. 2009. Visualizing Higher-Layer Features of a Deep Network. *Technical Report, Univeristé de Montréal*.

European Commission. 2021. Proposal for a regulation laying down harmonised rules on artificial intelligence.

Alex Foote, Neel Nanda, Esben Kran, Ioannis Konstas, Shay Cohen, and Fazl Barez. 2023. Neuron to Graph: Interpreting Language Model Neurons at Scale. ArXiv:2305.19911 [cs].

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2020. The Pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer Feed-Forward Layers Are Key-Value Memories. ArXiv:2012.14913 [cs].

Aaron Gokaslan and Vanya Cohen. 2019. Openwebtext corpus. `http://Skylion007.github.io/OpenWebTextCorpus`.

Wes Gurnee, Neel Nanda, Matthew Pauly, Katherine Harvey, Dmitrii Troitskii, and Dimitris Bertsimas. 2023. Finding Neurons in a Haystack: Case Studies with Sparse Probing. ArXiv:2305.01610 [cs].

Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. 2022. Mass-Editing Memory in a Transformer. ArXiv:2210.07229 [cs].

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv:1301.3781*.

Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. 2018. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73:1–15.

Neel Nanda. 2022a. NeuroScope.

Neel Nanda. 2022b. Transformerlens.

Neel Nanda. 2023. Actually, Othello-GPT Has A Linear Emergent World Representation. *Alignment Forum*.

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. 2023. Progress measures for grokking via mechanistic interpretability. ArXiv:2301.05217 [cs].

Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. 2020. Zoom in: An introduction to circuits. *Distill*. Https://distill.pub/2020/circuits/zoom-in.

Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. 2017. Feature Visualization. *Distill*, 2(11):e7.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. 2022. In-context learning and induction heads. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html.

OpenAI. 2023. Gpt-4 technical report.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library.

Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. 2022. Grokking: Generalization Beyond Overfitting on Small Algorithmic Datasets. ArXiv:2201.02177 [cs].

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Vikas Raunak and Arul Menezes. 2022. Rank-One Editing of Encoder-Decoder Models. ArXiv:2211.13317 [cs].

Ludwig Schubert, Michael Petrov, Shan Carter, Nick Cammarata, Gabriel Goh, and Chris Olah. 2020. OpenAI Microscope.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. ArXiv:1312.6034 [cs].

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention is all you need.

Jesse Vig. 2019. Visualizing attention in transformer-based language representation models. *arXiv:1904.02679*.

Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. 2022. Interpretability in the Wild: a Circuit for Indirect Object Identification in GPT-2 small. ArXiv:2211.00593 [cs].

Xuena Wang, Xueting Li, Zi Yin, Yue Wu, and Liu Jia. 2023. Emotional intelligence of large language models. *arXiv:2307.09042*.

Shijie Wu, Ozan Irsoy, Steven Lu, Vadim Dabravolski, Mark Dredze, Sebastian Gehrmann, Prabhanjan Kambadur, David Rosenberg, and Gideon Mann. 2023. Bloomberggpt: A large language model for finance. *ArXiv*, abs/2303.17564.

Catherine Yeh, Yida Chen, Aoyu Wu, Cynthia Chen, Fernanda Viégas, and Martin Wattenberg. 2023. AttentionViz: A Global View of Transformer Attention. ArXiv:2305.03210 [cs].

A Limitations

Mechanistic interpretability for practical usage in safety monitoring of large language models (LMMs) is limited in its applicability due to the large amount of neurons in state-of-the-art LLMs such as GPT-4 (OpenAI, 2023) and the difficulty in reverse-engineering the learned functions. DeepDecipher does not directly address this issue, but since it provides easy access to data on many neurons, it can hopefully aid the process regardless.

All three methods rely on max activating examples. The neurons' behaviour here may not be representative of its behaviour in general. This is strictly speaking not a limitation of DeepDecipher but rather of the methods themselves, but this points to the broader point that since DeepDecipher does not come with any novel methods, it is limited by the state of current LLM MLP interpretability methods.

The usability of DeepDecipher in practice is currently broadly untested due to time constraints. This is high on the list of next steps as it is necessary to guide further improvements.

B Related works

Mechanistic interpretability: (Olah et al., 2017) and (Olah et al., 2020) introduce mechanistic interpretability as the foundational pursuit to reverse-engineer the algorithms learned by neural networks. When (Elhage et al., 2021) defined a mathematical overview of the Transformer models, mechanistic interpretability became more focused on language models (Nanda, 2023; Elhage et al., 2022a,b). Existing examples of research work within Transformer mechanistic interpretability includes finding functional circuits for constrained semantic tasks in large language models (LLMs) (Wang et al., 2022) and taking steps towards automating this process (Conmy et al., 2023), using mechanistic interpretability to define metrics for grokking in LLMs (Nanda et al., 2023; Power et al., 2022), and defining a new softmax linear activation unit (Elhage et al., 2022a) to avoid the issues of semantic superposition and polysemanticity present in transformer models (Elhage et al., 2022b). (Foote et al., 2023) and (Bills et al., 2023) formulate techniques to build models of an MLP neuron's activation based on the augmented training examples that neuron activates the most for. (Geva et al., 2021) find that semantic explanations of neuron activity correspond with that neuron's activity, (Gurnee et al., 2023) find different groups of neuron activity, and (Antverg and Belinkov, 2022) critique linear probes and propose new models for analyzing MLP neurons.

Research tooling for mechanistic interpretability: Contemporary interpretability research has two challenges that tools for interpretability can solve; 1) to extract the features and variables we need for research, we often have to rerun models, which can be slow and expensive, and 2) accessibility of large language model internals (Schubert et al., 2020). (Schubert et al., 2020) introduce the OpenAI Microscope, a tool that visualizes the maximum activating examples for layers and neurons in eight vision models. Inspired by the OpenAI Microscope, (Nanda, 2022a) introduces NeuroScope, a website with access to all MLP neurons' most activating dataset examples for 25 Transformer models. NeuroScope is built using TransformerLens (Nanda, 2022b), a Python package that interfaces with PyTorch (Paszke et al., 2019) to extract activations on input from Transformer models. (Yeh et al., 2023) introduce a visualization technique for attention that uses query-key embeddings to understand and visualize self-attention mechanisms in Transformers.

Model editing: Mechanistic interpretability seeks to reverse-engineer neural networks to ensure that we understand them before we deploy them in real-world scenarios (Olah et al., 2017; Amodei et al., 2016). Techniques exist to update the activation behavior of targeted neurons, e.g., data-based model intervention by fine-tuning or model retrainings along with direct intervention methods, such as ROME (Raunak and Menezes, 2022), and MEMIT (Meng et al., 2022). These methods identify neurons associated with specific semantic associations and perform direct intervention on the activation through ablations (Durrani et al., 2020)

C API usage

Our repository includes a detailed and up-to-date description for how to use DeepDecipher. However, we provide a short description of the core usage here. All presented endpoints should be prepended with `https://deepdecipher.org`.

Querying: Queries use the format

```
/api/<model>/<service>/<layer>/<neuron>
```

This returns a JSON object with information specified by `service` (e.g. `neuroscope` or `neuron2graph`) for the specified model, layer, and neuron. For example

```
r=requests.get('https://deepdecipher.org/api/solu-8l/neuroscope/7/1423')
```

returns a JSON object with all the Neuroscope data for the 1423rd neuron in the 8th MLP layer of the `solu-8l` model. One then only needs to write

```
snippets = [r.json()["data"]["texts"][k]["tokens"]
        for k in range(10)]
```

to get the tokens of the first 10 texts for the neuron. Similarly, querying the layer and model information follows a similar syntax:

```
/api/<model>/<service>/<layer>
/api/<model>
```

Here is an example using the neuron store API to `neuron2graph-search` that receives a trimmed and lowercase token and returns lists of neurons that (a) activate the most to that token and (b) whose activation is most affected by this token.

```
/api/solu-6l/neuron2graph-search?query=any:the
```

Here, the response will be all neurons that match either (a) or (b) for token `the` since we use the `any` keyword. In this example, we receive 1976 results. For comparison `he` returns 254 neurons, `she` return 126 neurons, and `dream` returns 4 neurons in the following format.

```
[
    {layer: 2, neuron: 1917},
    {layer: 5, neuron: 2799},
    ...,
    {layer: 5, neuron: 734}
]
```