Measuring Monosemanticity via Causal Scrubbing

Authors	Brian Huang
Team Lead	
Code	
Description	We can measure how monosemantic a neuron really is via causal scrubbing
Poster	monosemantic.pdf

Histogram and correlation methods for monosemantic neuron detection
Neuron firings across text, bucketed into activation ranges (Figure 2)
Correlation with perfect binary neuron (fires 1 on concept words, 0 otherwise)

Abstract

Previous work has attempted to interpret the MLP hidden layers in transformer models such as GPT-2 small through the lens of monosemanticity. Monosemantic neurons, whose representations are dominated by a single feature corresponding to a single semantic concept in text, may provide an important pathway for mechanistic interpretability of a model's computation on natural language. However, there is evidence that monosemantic neurons may be much less common than previously thought, and polysemanticity of a neuron is difficult to rule out. In this project, we shift our viewpoint from detecting monosemantic neurons to deducing the degree of monosemanticity that a neuron has. A neuron's represented features may comprise a combination of multiple semantic concepts, and it is meaningful to ask how much of a neuron's latent space is devoted to any single concept. We propose a few quantitative methods for finding top candidates for monosemantic neurons across a transformer model given a semantic concept, which reduces the degree of hand-evaluation bias involved in detection. Furthermore, we propose a causal scrubbing setup with

alternative final metrics which potentially provide a concrete measure for degree of monosemanticity of a neuron.

Introduction & Background

Transformer architectures, in which attention heads and MLP units comprise the vast majority of parameters, are crucial to state-of-the-art performance on a wide range of natural language tasks. As such, a significant portion of mechanistic interpretability efforts for analyzing transformers' computations have focused on understanding MLP units. There are core differences between attention heads and MLPs through the interpretability lens. While MLPs cannot directly attend to previous tokens beyond the information encoded in previous attention OV circuits and in the residual stream, MLPs contain the only nonlinear activations in a transformer architecture. The presence of nonlinear activations hint at natural language functionalities of MLPs such as semantic understanding and memory storage/recall, as elucidated in previous research.

One route for interpreting MLPs hinges on the view that neurons represent distinct semantic concepts. (Elhage et al., 2022a) argued that nonlinear activations impose a *privileged basis* on neurons, in which neurons are encouraged to learn features that align with this basis in representation space; in the case that a neuron learns a single feature in the absence of superposition and that this feature corresponds to a single semantic concept, the existence of nonlinear activations then provides a mechanism for monosemanticity. ROME, the model editing method in (Meng et al., 2022), hinges on an ansatz that MLP units in transformers such as GPT-2 function as key-value correspondences, in which the hidden layer encodes representations of input text into a key which the output layer then uses for memory recall. This formulation of transformer MLPs as key-value memory units was earlier developed in (Geva et al., 2021), in which the key encoding functionality is subdivided among each individual neuron in the MLP hidden layer, (Geva et al., 2021) argued that single MLP neurons are statistical and/or semantic pattern detectors for input text; MLP output layers convert these pattern detections into votes on the output logit distribution, which are aggregated across the residual stream.

While these previous works provide crucial frameworks for understanding MLP hidden layer neurons (and in turn, MLP functionality), the analysis of semantic concepts in neuron representations is more difficult in practice. In (Elhage et al. 2022), to verify whether neurons are monosemantic, the authors employ hand evaluators to view a

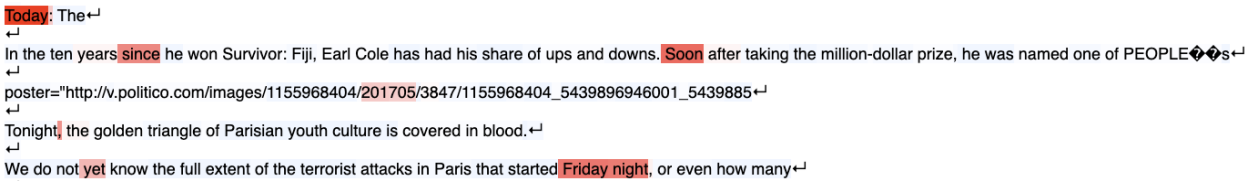
neuron's firings across a sample of text and mark the neuron as monosemantic if they can attribute 80% of the firings to a single semantic concept. This method of human evaluation introduces various biases—such as not rigorously accounting for the magnitude of neuron activations, and judging based on a limited number of tokens, a small fraction of the 50257 total tokens in GPT-2's vocal size—that limit our level of confidence that a monosemantic neuron found via hand-evaluation is truly monosemantic. Indeed, previous work on an "interpretability illusion" in LMs (Bolukbasi et al., 2021) found that in many cases, neurons which appeared to only fire on one semantic concept for a given dataset would then fire on unrelated semantic concepts when introduced to unseen or out-of-distribution data. Given these hand-evaluation and "interpretability illusion" constraints, it may be very difficult to disprove representation of additional semantic concepts in a neuron which initially appears to be monosemantic.

Although (Elhage et al., 2022b) indeed found single-feature neurons for a toy example of ReLU MLPs on a vector projection task, it is unclear whether the same mechanism for neurons learning to represent single features persists in larger-scale settings of transformers trained on a large natural language corpus. In addition, it's unclear whether the features learned in these settings correspond to single semantic concepts, or if they may encode more "non-robust," human-inscrutable aspects of natural language text. In our initial hand-evaluations of GPT-2 small, we found that candidate neurons for monosemanticity would still sparsely fire on tokens unrelated to the semantic concept we attributed to the neuron. As such, for the setting we are studying, it may be possible that "perfect" monosemanticity is very rare. To more generally interpret neurons that we believe to be monosemantic, it is therefore less meaningful to ask whether they are perfectly monosemantic vs. actually polysemantic, and more meaningful to ask to what degree they are monosemantic. A neuron is able to propagate certain information downstream to its MLP output and the residual stream, and this information can encapsulate a combination of statistical information and/or semantic concepts; can we figure out what percentage of this information is devoted to a given semantic concept?

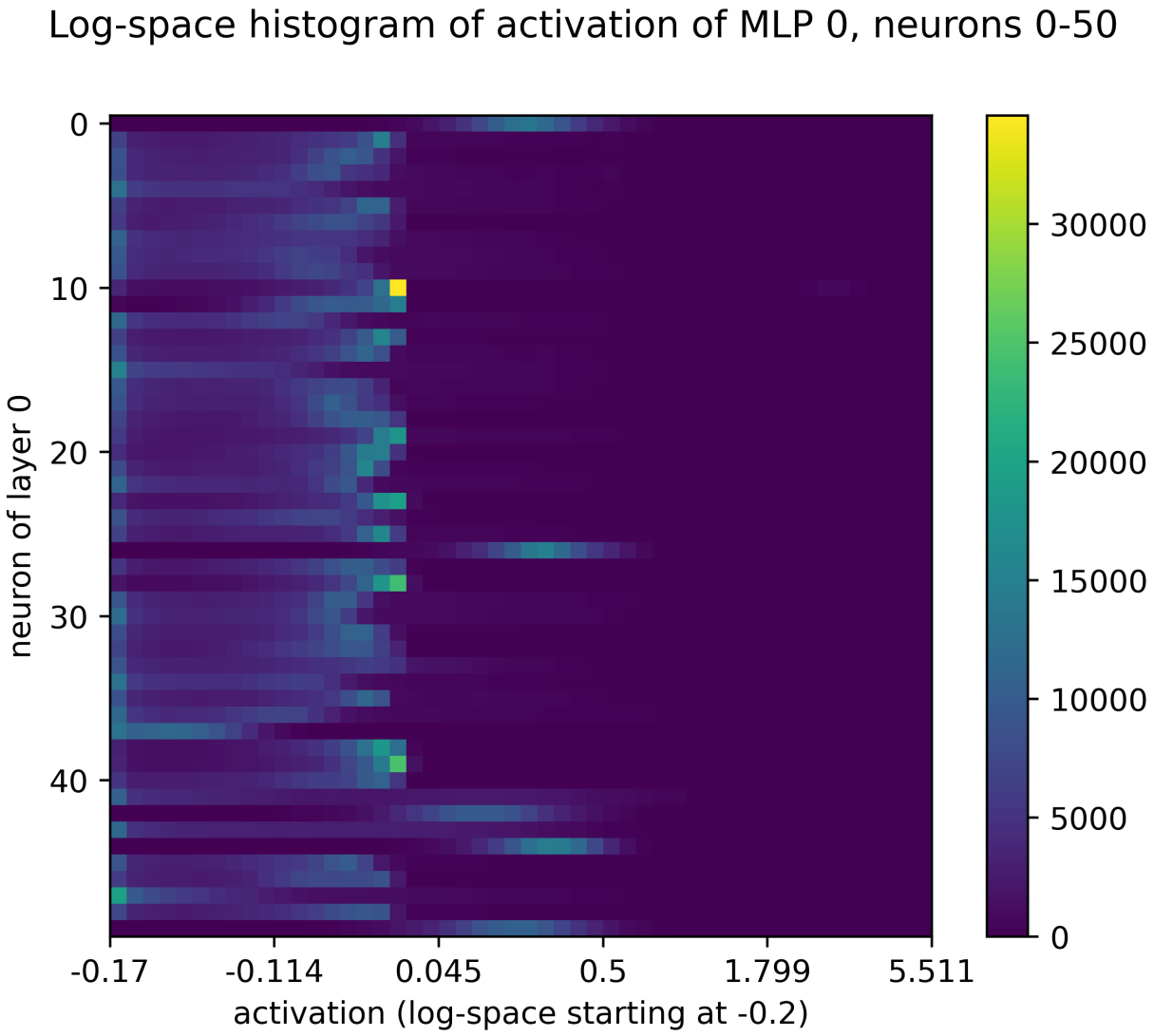
Detecting Monosemanticity

We first discuss a few systematic methods we used to survey all MLP hidden layer neurons in a transformer for monosemanticity, in order of less and less hand-evaluation required. Before answering the downstream question of "to what degree is a neuron

monosemantic for some semantic concept?", we'll need to detect and single out some strong candidates for monosemantic neurons. For this task, manually reading neuron activations from text samples provides a solid starting point. We construct a visualizer for displaying these neuron representations. Various examples of at least partial monosemanticity can be gleaned right off the bat from the visualizer; below, we show a neuron which fires almost exclusively on the time-related words in a random text snippet.



Another monosemanticity detection method involves evaluating neuron activation values across a large random subset of text samples, and bucketing the values into a range of intervals. This "histogram" technique shows us how often a neuron fires at each of various magnitudes. A monosemantic neuron would expect to fire at high magnitudes of activation on a small percentage of tokens, and not activate, i.e. fire at magnitude very close to zero, on "most" tokens. As such, our histograms aren't as helpful for directly finding monosemanticity, but they can help us rule it out for many neurons. For example, many neurons we found through hand-evaluations appeared to be generally excitatory or inhibitory neurons, with uniformly positive or negative activations, respectively, on every token across a large sample of text. Clearly these neurons cannot be monosemantic. In the histogram, such neurons will have their firings spread out across a region of only positive or only negative intervals, with no presence at the zero bucket. For example, this is what we see for the majority of the first 50 neurons in the layer 0 MLP of GPT-2 small:



Finally, given a “context-independent” semantic concept for which all single tokens can either be whitelisted as being part of the concept or not, we detect monosemantic neurons with a correlation technique. Using the whitelist for our semantic concept, we construct a binary neuron that fires with activation value 1 on any token in the whitelist, and conversely fires with activation value 0 on any token not in the whitelist. This binary token approximates the behavior of a perfect monosemantic neuron for our given concept. For each MLP neuron in our transformer, we compute the correlation between the MLP neuron’s firings on a large random sample of text and the perfect binary neuron’s firings on the same text; the neurons with the highest correlation scores can then be selected as candidates for monosemanticity on our concept. We remark that

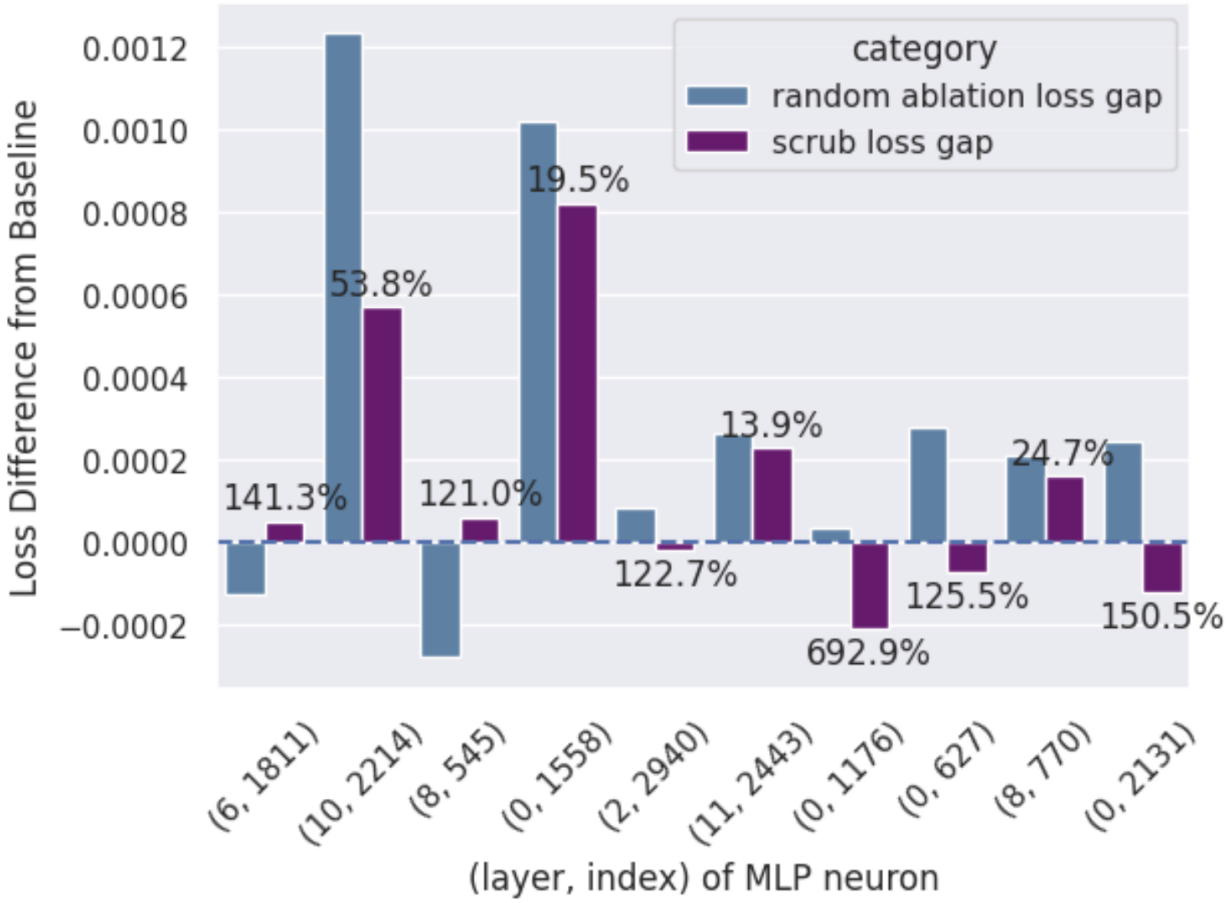
this correlation score for a neuron can already be interpreted, informally, as a rudimentary measure for degree of monosemanticity. As such a measure, the correlation is highly imperfect (for example, the firing values of an MLP neuron on tokens which are part of its semantic concept will clearly not just be 1), but there is room for refinement here.

In the next experiment, we study one specific semantic concept, that of prepositional words. The top ten prepositional word neurons are detected from the correlation method and used in our causal scrubbing formulation.

Causal Scrubbing

Recall that, in all three of our methods for detecting monosemanticity of neurons, we assume that perfect monosemantic neurons should fire in two distinct buckets of activation values; a nonzero range for tokens in the semantic concept, and in the range close to zero for tokens not in the semantic concept. This naturally leads to a causal scrubbing hypothesis for testing monosemanticity. We split the neuron by position and then resample-ablate the inputs to each position of the neuron. For each position of the ablated neuron, if the original text input to the model has a token at that position which is in the concept whitelist (resp. not in the whitelist), then that position slice of the neuron receives a random text input from the corpus whose token at that position is also in the concept whitelist (resp. not in the whitelist). If the neuron is monosemantic and fires like a monosemantic neuron, the scrubbed inputs should result in the same neuron activation values as before, and therefore the same overall model computation as before. This *scrubbed* setup is compared to the *original* setup, where no ablation is performed, and a *random* setup, where the entire neuron receives an unconditioned random input from the corpus.

We perform this causal scrubbing experiment on the top ten preposition word neurons. We initialize our GPT-2 small instance with a sequence length of 9, and perform inference on 10000 random text samples from openwebtext. For the final metric, we first use loss:

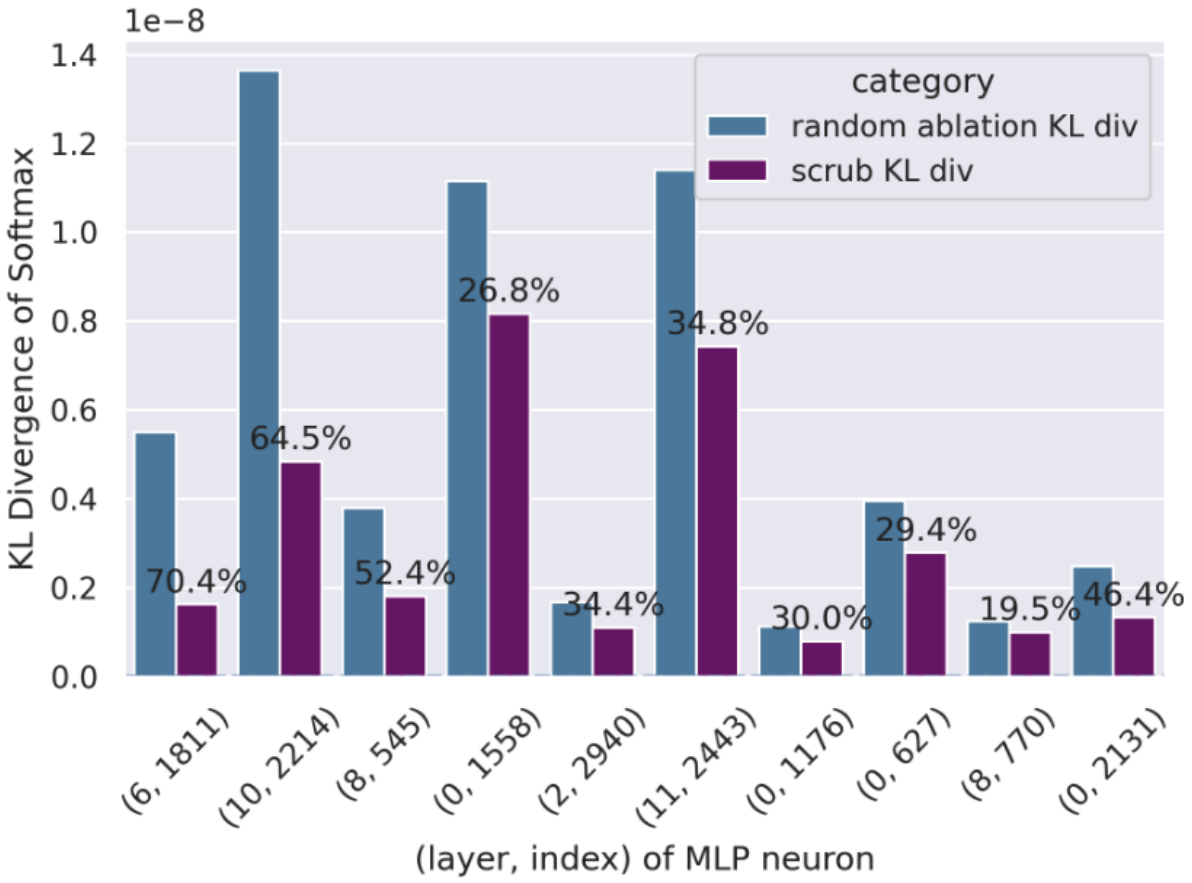


Across the ten neurons, the random ablation and causal scrub ablation lead to inconsistent behavior in the change in loss. In every case, the causal scrub exhibits a positive percent loss recovered (displayed on top of every scrub loss in the figure). However, the random ablation loss differences, scrubbed loss differences, and percent loss figures vary wildly and don't seem to correlate with the relative degrees of monosemanticity between the neurons hinted at by the correlation method. More egregiously, the 1811th neuron of the layer 6 MLP as well as the 545th neuron of the layer 8 MLP actually lead to a decrease in loss, in which cases the "positive percent loss recovered" is a misnomer—the scrub actually increases loss relative to the random ablation!

Because the loss is only measured based on the winning logit in our output distribution, our ablations provide some evidence that a neuron's computation on the semantic concept we attribute to it actually has an ambiguous effect on the winning logit. As such, we shift our focus to another metric, the KL divergence between the logit softmax of our ablated model and that of the original model. Even if perturbing a monosemantic neuron

doesn't influence the one winning logit in any specific direction, we can be more confident that it perturbs the overall distribution of logits. As such, KL divergence as a final metric may illuminate more about how a random ablation of a monosemantic neuron perturbs the logit distribution away from the original, and how scrubbing the neuron may "recover" the perturbed distribution to be closer to the original.

We show our causal scrubbing results with the KL divergence below:



As we predicted, the KL divergence metric behaves much more consistently. Compared to random ablation of a neuron, scrubbing that neuron significantly decreases the KL divergence for that neuron, and this pattern holds across all 10 neurons. We also see that "percent KL divergence recovered" is highest for the top 3 preposition neurons. While the relationship between percent KL divergence recovered and ranking of monosemantic neurons here is too noisy (and samples are too few) to conclude any positive correlation, we see some hints that KL divergence between softmax logits can be an appropriate measure of degree of monosemanticity of a neuron.

Discussion

The inconsistency in our causal scrubbing loss metric actually agrees with several aspects of the MLP key-value formulation in (Geva et al., 2021). After a neuron performs pattern detection for semantic concepts in the input representation, it passes this information onto the “value neurons” of the MLP second layer, and the resulting output is aggregated with the residual stream, which includes the outputs of earlier MLPs. Essentially, there is an arbitrary linear transformation before a neuron’s extracted information on a semantic concept is outputted into the residual stream, and then arbitrary “interference” between that MLP’s output with other layers’ MLP output. While a monosemantic neuron’s activations may provide a directional “vote” for tokens of that semantic concept, the final direction and effect of that vote depends highly on the corresponding MLP’s output matrix, as well as the behaviors of other MLPs on the same semantic concept. For example, (Geva et al., 2021) find frequent instances of input text where the final winning logit is distinct from any of the top *n* logits at each MLP output for some *n*. Even if a neuron activation on a given semantic concept does propagate to the MLP’s top few logits after processing by its output matrix, the MLP’s information about that semantic concept still may not necessarily have a consistent effect on the model’s final winning logit.

On the flip side, it’s possible that our KL divergence-based metric can give a consistent and even calibrated measure of how much of the information that a neuron extracts from input text, on average, is devoted to a single given semantic concept. Consider the case of perfect monosemanticity, i.e. a neuron only extracts information about a single semantic concept; here, the neuron will output certain activation values for each word that belongs to the concept, and output separate throwaway values for any word not in the concept, interchangeable between such non-concept words. On this neuron, if we perform a causal scrub which randomly resamples words not in the concept, the resulting neuron outputs will be identical to the original in expectation, resulting in zero KL divergence between the scrubbed and original models’ logits. (The losses would also be identical, but recall from our previous argument that we don’t know purely from looking at losses if we may have a false positive; lesser degrees of monosemanticity can produce zero loss difference.) If our neuron is not perfectly monosemantic and also extracts information about unrelated semantic concepts or statistical patterns, then our scrubbed neuron activations would deviate from the original activations, and this deviation would propagate to differences in final logits captured by KL divergence.

We don’t have a formal argument for how calibrated the KL divergence measure may be, but the intuition is somewhat there. After the neuron’s activation passes through the MLP hidden layer nonlinearity, the pathway from “value neurons” → MLP output → residual stream → pre-softmax output logits is entirely linear, so the neuron’s deviation from perfect monosemanticity on a concept is carried to the final output through a pathway of magnitude-respecting operations. There are various confounding factors here, such as the nonlinearities of any pathway from MLP output to residual stream that reenters later attention heads and MLPs; whether the final softmax linearity warps our attempt at calibration; or whether the scrubbing setup of KL divergence difference with a random ablation vs. scrubbed setup is the proper comparison to make for KL divergences. At the very least, it’s worth investigating further.

Further Directions

Building directly from previous discussion about the calibrated-ness of the KL divergence measure, we would want to further test that idea empirically. The correlation method for finding monosemantic neurons actually doubles as a rudimentary measure of degree of monosemanticity, so for a larger variety of semantic concepts for whether the “KL divergence recovered” does generally decrease as we go from higher-ranked to lower-ranked neurons for that semantic concept.

Furthermore, KL divergence doesn’t only apply to the output logits. The output representation of any MLP can be unembedded and softmaxed into a logit distribution. Therefore if we want to localize our scrubbing measurement closer to the neuron instead of at the final output, it’s justifiable to use “KL divergence difference” of the unembedded MLP outputs. (Geva et al., 2021) argue that MLP outputs indeed function as “votes” on the output distribution, providing evidence that the model interprets MLP outputs as softmax probability distributions. (Is there a difference between a summand for a probability distribution (a “vote”) vs. being a probability distribution by itself that we should be careful about?) It’s worth further investigating this claim about the functionality of MLP outputs, and scrubbing for “KL divergence difference” on umbedded MLP outputs might give us more insight into its salience.

The most localized scrub possible would be to use neuron activation as the final metric. We’d be able to verify whether neurons activate in distinct buckets of values depending

on the input's belonging to the semantic concept; find and verify more fine-grained distinctions between activation values for different words in a concept; and how position-independent and/or context-independent a neuron's activations are. The tradeoff with going from global (model-wide) to local (neuron-specific) metrics is that we get more direct verification of a neuron's degree of monosemanticity and its mechanisms for extracting semantic information, but we don't get a view of how the model as a whole uses this neuron for computation on this semantic concept. Performing scrubs with various final metrics from local to global allows us to explore both sides of the spectrum.

With our focus on a single neuron's degree of monosemanticity for a given concept, a natural next step is to ask whether we can find multiple semantic concepts that a neuron operates on, and get a sense of "how much" of a neuron's total information is, on average, devoted to each concept, or to the set of concepts (if individual concepts interact/interfere). The ideal goal here could be to find *all* semantic concepts that a neuron operates on, and compose them into a single scrubbing hypothesis that proves "perfect polysemanticity," where we might see, say, zero KL divergence at the output logits or unembedded MLP outputs.

Beyond Causal Scrubbing

Causal scrubbing is very powerful for examining all kinds of downstream effects of a single neuron's computation on a semantic concept, but it's highly constrained for studying monosemanticity beyond a single neuron. [For example, the scrubbing library actually does not permit performing our scrub on multiple neurons, because the conditional sampling tree becomes a DAG.] One important next step here is to experiment on the entire set of neurons in a model that are highly monosemantic for a given concept. As such, we'd be interested in exploring model-wide semantic information for a given concept through the lens of subspaces. Thanks to the subspace solver by Kai Jia [a fellow REMIXer!], we can isolate the subspace that a neuron operates on through its associated MLP output; if the neuron is highly or even perfectly monosemantic on some semantic concept, the resulting subspace corresponds to the MLP's computation on that semantic concept via that neuron. We can investigate if the MLP operates on the same subspace for each of the highly monosemantic neurons it contains for a given concept. Furthermore, we can investigate if different layer MLPs in our transformer operate on the same subspace for a semantic concept. One crucial element of these MLP-wide subspace investigations is developing an appropriate

similarity measure between the subspaces extracted by the solver. Way down the line, having a solid understanding of how any given semantic concept may map to a certain subspace across the different layer MLPs of a transformer can potentially open up new model editing methods.

References

Tolga Bolukbasi et al. “An Interpretability Illusion for BERT” (2021)

Nelson Elhage et al., “Softmax Linear Units” (2022a)

Nelson Elhage et al., “Toy Models of Superposition” (2022b)

Mor Geva et al., “Transformer Feed-Forward Layers Are Key-Value Memories” (2021)

Meng et al., “Locating and Editing Factual Associations in GPT” (2022)