Semantically Interpretable Neuron Clusters in BERT

Heidi Zhang
Stanford Univerisity
chenyuz@stanford.edu

Chenxi Gu
Stanford Univerisity
chenxigu@stanford.edu

Violet Yao
Stanford Univerisity
vyao@stanford.edu

Abstract

Human brains are known to contain specialized functional groups, and prior work in neural network interpretability has also found evidence of such expert groups. In this work, we propose an unsupervised approach for exploring clusters of language model neurons that specialize in various semantic concepts, using only general-purpose text corpora. We take token activation values of neurons in BERT feed-forward layers, process the activations into neuron representations, and apply hierarchical clustering to find specialized neuron clusters. We observe that for many clusters, the top-activating tokens show semantically meaningful patterns. We evaluate the effectiveness of found clusters by intervening neurons in the clusters and comparing performance drop with random baselines, and conclude that with an appropriately small size of the clusters (e.g. around 50 neurons in each), each cluster indeed contributes above-random to their associated semantic pattern.

Cluster ID: C4-16/200
Theme: Positive achievements and hopes
Top-activating Tokens: [‘promising’, ‘praying’, ‘preliminary’, ‘desirable’, ‘importance’, ‘presumably’, ‘testament’, ‘pioneering’, ‘carefully’, ‘revelation’]

Cluster ID: Yelp-165/200
Theme: Time and routines
Top-activating Tokens: [‘monday’, ‘saturday’, ‘weekday’, ‘homework’, ‘thursday’, ‘friday’, ‘tomorrow’, ‘##sr’, ‘else’, ‘wednesday’]

Table 1: Top-activating tokens for two example clusters from the C4 and Yelp datasets. Both are from the 200-cluster experiment with smoothed representations (Section 6). Cluster themes are abstracted with the help of ChatGPT.

1 Introduction

While complex transformer-based language models are often treated as black boxes, it is essential to understand their internal mechanisms in order to gain insights into their decision-making process. Previous work has looked at model interpretability at varying levels of granularity, including the model behavior level, layer or representation level, and, the most granular, the level of individual neurons or specific input features. In this work, we start from the neuron level and build up neuron clusters in an unsupervised way, to find interpretable patterns that contribute to the inner workings of transformer-based language models.

Our hypothesis that neurons may form interpretable clusters comes from the idea of modularity, or functional regions, of the brain. Modularity theory suggests that complex neural networks can be composed of modular subnetworks which are "semi-independent units that perform specific functions or computations within the larger network". The benefits of modularity include reusability, scalability, and interoperability.

While it may be impossible to find perfectly modular subnetworks in today’s large language models, due to the way they are constructed, we aim to find modular patterns (or experts) that specialize in different semantic roles. We use tokens that a neuron activates on as a reflection of their semantic role and find modules by clustering neurons based on their token-level activations. We then look for human-interpretable explanations for the modules using their top-activating tokens.

In order to evaluate the quality of the clusters that we find, we apply causal intervention to the model by ablating neurons in each cluster, and compare the drop in masked-language-modeling (MLM) performance with random ablation baselines, targeted at the cluster’s top-activating tokens. Higher-than-random performance drop would indicate that the found cluster plays an essential role in those tokens.

We discover semantically meaningful neuron clusters in the feed-forward layers, where each cluster focuses on relevant concepts interpretable at the token level (Table 1). We find that for a cluster to be reduced to a single topic, such as "month" or "city/state", a large number of clusters (and thus a small number of neurons in each cluster) is needed, and some neurons simply cannot fit into an interpretable cluster. However, for the clusters that we find, they show interesting activation patterns for related semantic concepts, and they also show a unique positional pattern across transformer layers. Evaluation results show that, with the correct neuron representation and cluster number, the clusters we find are indeed specialized submodules for the concepts they represent.

2 Prior Literature

One recent work by OpenAI explores GPT-4's (OpenAI, 2023) ability to generate explanations of individual neuron activations (Bills et al., 2023). They use (token, activation) pairs as input to GPT-4, the explainer model, to obtain explanations for each neuron. To verify the explanations, they use a simulator model (also GPT-4) to simulate the neuron's activation based on the explanation. The quality of the explanation is assessed by comparing the simulated activation to the actual activation of the neuron. The authors find reasonable explanations and interesting patterns, such as specific neurons for similes, certainty and confidence, and pattern breaks. However, challenges remain as many neurons are poorly explained. Additionally, the complex nature of neuron behavior and the potential collaboration of multiple neurons in circuits may hinder concise summarization.

Corpus-based neuron-level explainability approaches provide explanations for individual neurons by "aggregating statistics over data activations" (Bills et al., 2023; Kádár et al., 2017; Na et al., 2019). Causation-based methods, on the other hand, pre-define a set of concepts or knowledge-based tasks and identify neurons that encode those concepts or knowledge (Mu and Andreas, 2020; Dai et al., 2022). Neuron clustering methods employ an unsupervised approach to discover clusters of neurons based on their activations (Meyes et al., 2020; Dalvi et al., 2020).

While corpus-based and causation-based methods primarily analyze neurons individually, our focus lies on neuron clustering methods that examine groups of neurons. Meyes et al. (2020) train custom networks on MNIST datasets and observe clear separability of the network for digit classes, particularly in later layers. Dalvi et al. (2020) utilize neuron clustering to investigate redundancy in transformer-based deep neural networks, demonstrating that neurons exhibiting similar behavior are redundant and can be pruned. Nevertheless, neuron clustering methods have limitations, such as being task-specific (Meyes et al., 2020) or lacking comprehensive explanations (Dalvi et al., 2020). In our work, we adopt a similar approach, utilizing neuron co-activations to measure similarity and employing agglomerative clustering to group similar neurons, while hoping to address these limitations.

Another line of research pertains to modularity in neural networks. Modularity is characterized by the correlation between strongly interconnected system components, or modules, and their respective functions (Ballard, 1986). Modularity research utilizes aggregated activations for neuron clustering, grouping co-activated neurons together (Zhang et al., 2021; Hod et al., 2021; Pochinkov, 2023).

Although evidence suggests the existence of modules across different network types, challenges remain in achieving a comprehensive and interpretable framework. For example, Zhang et al. (2021) successfully identify and prune experts, achieving high performance with fewer parameters, but lack human-comprehensible interpretations. Hod et al. (2021) fall short in identifying specific tasks performed by neuron groups, while Pochinkov (2023) struggle to distinguish between general and Python-specific coding abilities.

The studies by Zhang et al. (2021) and Pochinkov (2023) suggest that larger neural network models may exhibit greater modularity. Larger models exhibit sparser activations, with a smaller proportion of neurons being activated. This might indicate a more specialized activation pattern. However, these findings contradict the neuron-level analysis by Bills et al. (2023), where a downward trend in neuron explainability is observed as the model size increases. The discrepancy could be due to larger models utilizing groups of neurons to process individual concepts, making it more challenging to extract neuron-level specialization. Aggregating insights from a group of neurons may provide more informative results, which motivates our research in this direction.

3 Data

We use two datasets to compute neuron activations and conduct Masked Language Modeling (MLM) tasks.

The first dataset is the **Yelp review** dataset, comprising reviews about businesses from Yelp. It was extracted from the Yelp Dataset Challenge in 2015. We use the test split of this dataset, which contains 50000 examples consisting of 8598966 tokens.

The second dataset we employ is the **Colossal Clean Crawled Corpus (C4)** (Raffel et al., 2019), a dataset of English texts from the web. Texts in this dataset were collected from the data scraped by Common Crawl in April 2019 and underwent a cleaning process. We use the validation split of the "en" variant of this dataset, which has 364608 examples consisting of 103320883 tokens.

The Yelp dataset predominantly focuses on experiences and feedback regarding food and services, while the C4 dataset covers a wider range of topics. Thus, we use Yelp for a more targeted cluster search where the cluster explanations can be expected to fall under a certain range, and use C4 as a more general-purpose corpus that contains little inductive bias. We compare the outcomes of our method across these two datasets to explore the extent to which the dataset influences the interpretability of neuron clusters. See Table 2 for one example from each dataset.

One example from Yelp:
"After waiting for almost 30 minutes to trade in an old phone part of the buy back program, our customer service rep incorrectly processed the transaction. This led to us waiting another 30 minutes for him to correct it. Don't visit this store if you want pleasant or good service."

One example from C4:
"I thought I was going to finish the 3rd season of the Wire tonight. But there was a commentary on episode 11, so I had to rewatch Middle Ground with the commentary. Hopefully I can finish the season next weekend."

Table 2: Examples from Yelp and C4.

4 Model

Our main approach involves two stages: find modules by clustering neurons based on their token-level activations, and look for human-interpretable explanations for the modules using their top-activating tokens. We would be able to confirm our hypothesis of language model modularity if we can successfully find human-interpretable explanations for the clusters we find, and that they evaluate well on the metrics we define.

Our approach is based on two limitations of previous work that we hope to address in our project. First, few works consider groups of neurons and existing work had limited success. As discussed in Suau et al. (2020), many intricate concepts may have to be represented by a more complex expert, such as a module, rather than a single-neuron expert. Second, in current modularity methods, each module often lacks human-interpretable explanations, and in neuron-analysis methods, in order to find human-interpretable explanations, supervised tasks are often required. Thus, we aim to solve these two issues by exploring human-interpretable groups of neurons using unsupervised clustering methods.

5 Methods

We work with the BERT model (Devlin et al., 2019a), which is a relatively simple yet powerful model, and has a lot of prior work for comparison. We dissect the pretrained BERT$_{\text{BASE}}$ model by analyzing the output of the embedding layer and the outputs of all the transformer blocks after the feed-forward layers. For BERT$_{\text{BASE}}$, the output for each token at the above layers has size 768. We use the term "neuron activation" to refer to each single variable out of 768 variables in the output at each layer, following the definition of neuron in Dalvi et al. (2020). Because BERT$_{\text{BASE}}$ has 12 transformer blocks, the total number of neurons is $13 \times 768 = 9984$.

We use the following notations throughout the rest of this paper:

Notations

V: vocabulary size (30522 for the tokenizer of BERT$_{\text{BASE}}$)
t_v: the v-th token in the vocabulary for $v \in \{1, \ldots, V\}$
N: number of examples in the corpus
$\mathbf{x}_i$: the i-th example in the corpus for

$i \in \{1, \ldots, N\}$
$|\mathbf{x}_i|$: number of tokens in $\mathbf{x}_i$
$w_{i,j}$: the j-th token in $\mathbf{x}_i$ for $j \in \{1, \ldots, |\mathbf{x}_i|\}$
$\mathbb{1}$: indicator function
L: number of layers (13 for $\text{BERT}_{\text{BASE}}$)
H: hidden dimension (768 for $\text{BERT}_{\text{BASE}}$)
$A_{l,h}(w_{i,j}|\mathbf{x}_i)$: the activation of the h-th neuron at the l-th layer on $w_{i,j}$ in the context of $\mathbf{x}_i$ for $l \in \{1, \ldots, L\}$ and $h \in \{1, \ldots, H\}$
$R_{l,h}$: the vector representation of the h-th neuron at the l-th layer
$R_{l,h}[v]$: the v-th scalar in $R_{l,h}$ for $v \in \{1, \ldots, V\}$
K: number of clusters

5.1 Computing Neuron Activations and Representations

We would like to establish a relationship between each neuron and each token, which enables us to interpret individual neurons or clusters of neurons using tokens. To this end, we use the activation of a neuron on a token to represent the significance of the token to the neuron, with higher values indicating greater significance. As the activation is influenced by the context of the token, we average over all of its occurrences in the corpus.

Equation 1 defines the average activation of the h-th neuron at the l-th layer on token t_v, $R_{l,h}[v]$, for $l \in \{1, \ldots, L\}$ and $h \in \{1, \ldots, H\}$ and $v \in \{1, \ldots, V\}$:

$$R_{l,h}[v] = \frac{\sum_{i=1}^{N} \sum_{j=1}^{|\mathbf{x}_i|} A_{l,h}(w_{i,j}|\mathbf{x}_i) \mathbb{1}[w_{i,j} = t_v]}{\sum_{i=1}^{N} \sum_{j=1}^{|\mathbf{x}_i|} \mathbb{1}[w_{i,j} = t_v]} \tag{1}$$

5.2 Post Processing Neuron Representations

Token Filter Tokens occurring infrequently in our corpus can introduce noise into neuron representations. To address this potential issue, we employ a frequency threshold. Indices corresponding to tokens occurring less frequently than the threshold are excluded. Additionally, indices corresponding to special tokens [PAD], [CLS], [SEP] are also discarded. For the Yelp dataset, we set the threshold at 20, resulting in the retention of 10179 (33%) indices/tokens. In the case of the C4 dataset, we select a threshold of 400, preserving 15298 (50% of vocabulary size) indices/tokens.

Semantic Augmentation (Smoothing) One issue with our basic token-activation-based neuron clustering is that the semantic meaning of the tokens and the similarities between tokens are ignored, since each token's activation is treated individually as a dimension in our neuron representation. This is disconnected with our goal of finding interpretable neurons.

In order to find semantically meaningful neuron clusters, we would like to encourage neurons that activate on similar tokens to be clustered together. To do that, we implement the following *semantic smoothing* approach: for each token in the vocabulary, find its top-5 most similar neighbor tokens and their similarity scores using fastText embeddings[1] (Bojanowski et al., 2016), and add the discounted average of similarity score $\times$ activation of each of the top-5 neighbor tokens to the current neuron representation.

Let λ be the discount value, which we manually set as 0.5 in our experiments, and $v' = v_1, v_2, ..., v_5$ be the top-5 neighbor tokens. The approach can be summarized in the following formula:

$$R_{l,h}[v] += \frac{\lambda}{5} \sum_{i \in v'} sim(t_i, t_v) \cdot R_{l,h}[i] \tag{2}$$

By augmenting the neuron representation at each token with the representation values of similar tokens, we essentially smooth the neuron representations by bringing close representation values at synonym tokens, which would benefit our goal of finding semantically meaningful clusters. However, we note that since the activation values are manually edited in this approach, the clusters no longer accurately reflect the original model outputs, but are tuned toward our expectations. Still, the discovered clusters can be seen as successful if they evaluate well on causal intervention metrics.

TF-IDF We use a vocabulary-sized vector for each neuron, which contains aggregated statistics for each token in the vocabulary associated with that neuron. This is similar to the bag-of-words representation used in document analysis. However, the bag-of-words representation only considers word frequency and can include common words that don't contribute much to understanding. To address this, we apply TF-IDF post-processing, treating each neuron as a document. TF-IDF assigns weights to words based on their importance within the neuron and across the entire dataset, giving higher weight to rare and distinctive words. This

[1]fastText can handle token in addition to words.

approach helps prevent common tokens from dominating the neuron representation and enables us to identify distinctive tokens for cluster differentiation.

5.3 Clustering Neurons

After obtaining representations for each neuron, we normalize each representation to be a unit vector and cluster the neurons using *hierarchical agglomerative clustering* with complete linkage, following Dalvi et al. (2020). Dalvi et al. (2020) uses the absolute Pearson correlation between the raw (unaveraged) activations of the two neurons as their similarity measure, while we use cosine similarity, without absolute value, since we believe that negatively related neurons should not be clustered together. Thus, our dissimilarity measure is defined as $1 - cos\_sim(R_i, R_j)$. While we do not find significant differences between cosine similarity and Pearson correlation, more experiments could be done to find the best setting.

Tuning Number of Clusters The number of clusters, K, is a crucial hyperparameter in our method. As mentioned previously, we cluster 9984 neurons. When K is set to a small value, it is possible that neurons that are not similar to each other are grouped into the same cluster. In this case, it becomes challenging to extract commonalities from these neurons. On the other hand, if K is set to a large value, some clusters may only contain very few neurons, covering overly specific/narrow concepts.

Cluster Refinement Many of the clusters obtained through agglomerative clustering are still large, especially when using a smaller number of clusters such as 20 or 50. In such cases, even neurons within a cluster may activate on different topics, so the cluster would not be very representative of a central topic. Another motivation for cluster refinement or pruning is that not every neuron could potentially be part of an interpretable cluster.

To refine a cluster, we use an iterative refinement strategy, where at each step, one neuron that is the farthest away from the cluster center is removed, and the cluster center is recomputed. The distance is computed as the Euclidean distance between each target neuron representation and the average representation (cluster center). This is repeated until the maximum distance is smaller than a threshold (set as 0.6 in our refinement step).

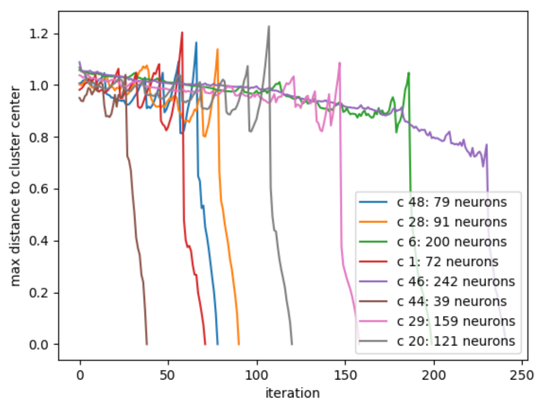


Figure 1: Max distance of neuron to cluster center after each refinement step, for 8 random clusters. Observe the substantial drop in the max distance when there are around 10 remaining neurons. Setup: C4, 50 clusters, filter-only.

Interestingly, we find that most clusters have a substantial drop in the maximum distance when the number of neurons is reduced to around 10 (Figure 1). This is consistent with the finding in Dalvi et al. (2020) that around 9/10 of BERT neurons can be safely pruned without hurting accuracy much.

Alternative refinement strategies may involve removing neurons that have low activations on the selected tokens that represent the cluster.

5.4 Explaining Clusters

Finally, we find human-interpretable explanations for each neuron cluster. We first compute the cluster representation by averaging the representation vectors of all the neurons in this cluster, and record the top-activating tokens in the cluster representation. The tokens are then used to interpret the cluster. We believe that the most positively activated tokens are representative of what a neuron is responsible for.

We further interpret the top-activating tokens by prompting large language models, such as ChatGPT or GPT-4, for a high-level explanation.

See Appendix A for an alternative approach of finding compositional explanations using a preselected concept bank.

5.5 Evaluating Clusters

Two important qualities that we want our clusters to have are representativeness and exclusiveness. Representativeness means that, within each cluster, neurons represent the same concept and are representationally close to each other. Exclusiveness means that each cluster represents a concept exclusively such that neurons from other clusters contribute significantly less to that concept. The following two methods focus on evaluating these two qualities respectively.

Cluster Closeness We evaluate the closeness within clusters, i.e. how closely the neurons within each cluster are related. To measure this, we utilize activation-based neuron representations to determine the representational proximity of neurons within a cluster.

Additionally, we visualize the embeddings of the top-activating tokens using fastText (Bojanowski et al., 2016) and project them onto a 2D space using t-SNE or PCA. This visualization helps us identify whether the top-activating tokens for each cluster can be well-separated and used as a semantically meaningful explanation. See Appendix B for embedding plots.

Additionally, we visually analyze the physical locations of neurons within a cluster, considering both the layer and the position on that layer, in order to identify any potential patterns in their distribution.

Causal Intervention We use causal intervention to evaluate the identified clusters. Our goal is to understand how turning off (ablating) neurons impacts the model's performance in the MLM task using the Yelp review dataset and C4 data. This approach is similar to the one used by Pochinkov (2023) to evaluate model separability. They focused on the next token prediction task since the model they examined was GPT-2 (Radford et al., 2019).

For this evaluation method, we need to identify or extract an appropriate small corpus from a large corpus to evaluate each neuron cluster. For each cluster, we select sentences that contain the top-activating tokens as the small corpus for evaluating that specific cluster. Our hypothesis is that removing the neuron cluster from the network will result in a decline in performance on the selected corpus while maintaining the same performance on other test corpora. Specifically, we compare the MLM loss of predicting the top-activating tokens with and without the cluster. To turn off neurons in a cluster, for the $\text{BERT}_{\text{BASE}}$ model (Devlin et al., 2019b), we set the activation of a neuron in the $i-1$th layer to the mean of the activations in that layer, passing it to the ith layer to generate the hidden states until the last layer. We then pass the last hidden states to a cls layer and compute the MLM loss using cross-entropy loss.

As a baseline, we compare the effect of randomly turning off the same number of neurons as those in the selected cluster. Additionally, each cluster may have varying numbers of neurons in each layer, and each cluster may exhibit distinct position patterns across the layers. Consequently, we establish three strong random baselines by randomly sampling neurons, randomly sampling based on the same layer distribution, and randomly sampling based on the position distribution as the selected cluster.

6 Results

6.1 Within-Cluster Distance

The number of clusters, K, determines the quality of clustering and the interpretability of each cluster. Ideally, neurons within a cluster should exhibit proximity to one another. To assess this, we introduce the *"max distance within cluster"* metric, which measures the largest distance between any pair of neurons within a cluster, and the *"average distance within cluster"*, which measures the average distance. They are then averaged over all clusters to provide an aggregated view. Figure 2 illustrates the relationship between the number of clusters and average max distance and average average distance. As expected, both distances decrease as the number of clusters increases. With 50 clusters, the average max distance is around 1.2 and the average average distance is around 0.8; with 200 clusters, the average max distance is around 1.0 and the average average distance is around 0.6. These experiments suggest that the distances within clusters are still relatively large when $K = 200$ (recall that we use 1 - cosine similarity as the distance metric, where a distance of 1 indicates a cosine similarity of 0 between the vector representations of two neurons). To avoid a large number of clusters, we decide to proceed with $K = 50$ and $K = 200$. Additionally, in terms of these two metrics, there is minimal difference between the post-processing strategies of "no filtering", "filtering only" and "filtering + smoothing".

6.2 Causal Intervention

Table 3 presents the results of causal intervention experiments comparing the MLM loss of predicting top-activating tokens with and without the clus-

Setup	Method	*cluster*	*random*	*random_layer*	*random_position*
C4 – 50	filter	5.33	**7.04**	6.54	0.13
	+ TF-IDF	5.20	**6.77**	6.52	0.30
	+ smoothing	4.01	5.66	**5.72**	0.10
	+ refine	**1.85**	0.23	-0.06	-0.02
C4 – 200	filter	**1.74**	0.80	0.82	0.16
	+ TF-IDF	**1.80**	1.46	1.33	0.20
	+ smoothing	**1.99**	1.13	1.23	0.12
	+ refine	**1.05**	0.27	0.16	0.00
Yelp – 50	filter	5.31	**6.76**	6.56	0.08
	+ TF-IDF	1.76	**5.86**	5.13	0.17
	+ smoothing	3.50	**6.59**	6.49	0.09
	+ refine	**1.98**	0.16	0.21	0.03
Yelp – 200	filter	**1.96**	1.48	1.30	0.12
	+ TF-IDF	0.65	1.20	**1.29**	0.16
	+ smoothing	**1.57**	1.23	1.23	0.14
	+ refine	**0.96**	0.20	0.28	-0.00

Table 3: Performance drops (%) for the causal intervention experiments, on two datasets (C4 and Yelp) and two clusterings (50 clusters and 200 clusters). The largest value for each row is highlighted in bold.

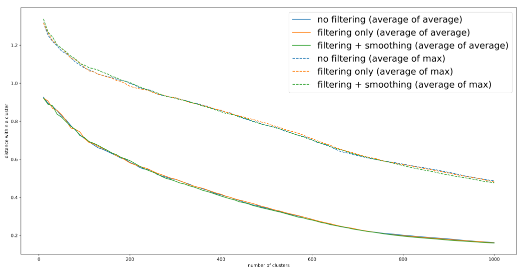


Figure 2: Average of "average distance within a cluster" and average of "max distance within a cluster" when grouping into different numbers of clusters. Setup: C4 (the plot for Yelp is similar)

ter, along with three random baselines. The experiments were conducted on two datasets, C4 and Yelp, using two different numbers of clusters: 50 and 200. The four methods evaluated for cluster identification are frequency filtering alone, frequency filtering plus TF-IDF, frequency filtering plus smoothing, and frequency filtering plus refine. The values in the *cluster* column represent the percentage change in MLM loss when the neurons within the respective cluster are turned off, relative to the original BERT model without neuron ablation. The random baselines, denoted as *random*, *random_layer*, *random_position* serve as comparison benchmarks. The *random* baseline randomly turns off the same number of neurons as those in the selected cluster. Since clusters may have varying neuron counts across layers, the *random_layer* baseline randomly turns off an equivalent number of neurons in each layer. Additionally, the *random_position* baseline randomly selects positions to deactivate across all layers, with the total number of neurons turned off matching the cluster's neuron count. This baseline is included because, when clustering neurons, we observe an index-clustering effect across all layers (section 7.3). We also provide a visualization of MLM loss for 50 clusters on the C4 dataset, comparing with and without the cluster, as well as the *random_position* baseline, in Figure 4 in the appendix.

The results indicate that the clusters identified by all our methods, when ablated, lead to a performance drop. With 200 clusters, we find consistently larger drops than all random baselines (except with TF-IDF in Yelp). With a smaller number of clusters, ablating our clusters has a smaller performance drop than random ablation and per-layer random ablation of the same number of neurons, but a much larger drop than ablations of neurons on selected positions across layers. The reason for this needs further investigation.

7 Analysis

7.1 Causal Intervention Analysis

We observe that our methods work better under causal intervention when the number of clusters is

Cluster ID: C4-43/200
Theme: Food and personal care
Tokens: [‘##oche’, ‘gi’, ‘cheese’, ‘protein’, ‘chi’, ‘##hee’, ‘barber’, ‘cot’, ‘peanut’, ‘hairs’]
Cluster ID: C4-42/200
Theme: Technology and business
Tokens: [‘cleanup’, ‘calculations’, ‘entrepreneurs’, ‘cookies’, ‘avengers’, ‘purchases’, ‘ipod’, ‘biscuits’, ‘ios’, ‘nerves’]

Cluster ID: C4-40/200-smoothed
Theme: Charitable activities
Tokens: [‘charitable’, ‘charities’, ‘charity’, ‘puzzles’, ‘simulations’, ‘tiled’, ‘vegetables’, ‘tiles’, ‘fundraiser’, ‘fundraising’]
Cluster ID: C4-116/200-smoothed
Theme: Technology and business
Tokens: [‘inspections’, ‘assessments’, ‘brakes’, ‘scanner’, ‘guards’, ‘inspection’, ‘motorcycles’, ‘sewer’, ‘kettle’, ‘brushes’]

Table 4: Top-activating tokens for clusters that score well in causal intervention. All are from the 200-cluster experiments; the first two use frequency filter only, and the latter two add semantic smoothing. Cluster themes are abstracted with the help of ChatGPT.

larger, i.e. when the number of neurons in each cluster is smaller.

Also, it is worth noting that after refinement, the number of neurons per cluster reduces to around 10, which is significantly smaller than the average number in our methods even with 200 clusters ($768 * 13/200 = 50$). As a result, changes in MLM losses are relatively smaller due to the reduced number of neurons in the refined clusters. Additionally, fluctuations in the random baseline values exist across methods due to the selection of evaluation corpus based on top activating tokens in a cluster.

7.2 Top-Activating Tokens

We show the top-activating tokens from clusters that have the largest difference between the MLM loss with “cluster turned off” and with the original model. Those are arguably clusters that focus more exclusively on the concepts they represent (Table 4), and qualitatively they seem well-explained.

7.3 Physical Positions of Neurons

To visualize the distribution of neurons within a cluster, we create scatter plots that depict the layer and location of neurons. Examples of these scatter plots can be found in Figure 5 in the appendix. It is intriguing to note that neurons in each cluster exhibit a fairly even distribution across layers, maintaining the same index positions in each layer. One possible interpretation of these results aligns with the findings in Dalvi et al. (2020), suggesting the presence of layer redundancy in BERT. Layers can be considered redundant if neurons at the same index positions across layers demonstrate similar functionality. Another interpretation is that neurons at each level retain positional information, leading neurons at the same index across layers to co-activate for the same token. Consequently, our co-activation-based clustering method groups these neurons together.

8 Conclusion

In this work, we proposed an approach to cluster neurons based on their token-level activations and interpret clusters using their top-activating tokens. We found small modules BERT neurons when the cluster number is large. Our causal intervention experiments for the clustering of 200 clusters demonstrated that the neurons within a cluster play a more significant role than randomly selected neurons in predicting the tokens they highly activate on when those tokens are masked. In terms of interpretability, some clusters highly activate on semantic concepts like “month” or “city/state”, while other clusters are less easily interpretable. Additionally, we observed interesting patterns in the spatial distribution of neurons within a cluster across different layers of BERT. Investigating the underlying reason would be an interesting extension of our work. Future works could involve alternative methods for generating neuron representations. For example, using feature attribution of neurons in well-designed tasks instead of activations. Moreover, as we have implemented TF-IDF vectors for each neuron, it may be promising to do “neuron retrieval” given input concepts or descriptions.

Known Project Limitations

In this section, we describe the limitations of this work. First, the neuron representations we compute only use token-level information, potentially losing contextual information that language models make use of. We would also be unable to find neuron interpretations that require sentence-level information, such as resolving coreferences or relating clauses. In addition, we only find explanations from the semantic aspect and ignore other potential aspects such as syntax or morphology.

We only work with neurons in the feed-forward layers of the transformer blocks, and future work may consider looking into the attention layers.

Finally, while we see reasonable causal intervention results with smaller clusters, the results are not very strong (only 1-2%) increase in loss when dropping the cluster. This indicates that finding interpretable clusters in language models is challenging due to the complex nature of those models.

Authorship Statement

Violet worked on causal intervention and TF-IDF. She also contributed to the visualization and analysis of results.

Heidi worked on representation filtering by frequent tokens and smoothing, the clustering algorithm, and cluster refinement.

Chenxi worked on computing neuron activations and exploring the relationship between within-cluster distances and numbers of clusters.

References

Dana H. Ballard. 1986. Cortical connections and parallel processing: Structure and function. *Behavioral and Brain Sciences*, 9(1):67–90.

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. `https://openaipublic.blob.core.windows.net/neuron-explainer/paper/index.html`.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. Knowledge neurons in pretrained transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8493–8502, Dublin, Ireland. Association for Computational Linguistics.

Fahim Dalvi, Hassan Sajjad, Nadir Durrani, and Yonatan Belinkov. 2020. Analyzing redundancy in pretrained transformer models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4908–4926, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019a. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019b. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Shlomi Hod, Stephen Casper, Daniel Filan, Cody Wild, Andrew Critch, and Stuart Russell. 2021. Detecting modularity in deep neural networks. *CoRR*, abs/2110.08058.

Ákos Kádár, Grzegorz Chrupała, and Afra Alishahi. 2017. Representation of linguistic form and function in recurrent neural networks. *Computational Linguistics*, 43(4):761–780.

Richard Meyes, Constantin Waubert de Puiseau, Andres Felipe Posada-Moreno, and Tobias Meisen. 2020. Under the hood of neural networks: Characterizing learned representations by functional neuron populations and network ablations. *ArXiv*, abs/2004.01254.

Jesse Mu and Jacob Andreas. 2020. Compositional explanations of neurons. *NeurIPS*.

Seil Na, Yo Joong Choe, Dong-Hyun Lee, and Gunhee Kim. 2019. Discovery of natural language concepts in individual units of cnns. *ArXiv*, abs/1902.07249.

OpenAI. 2023. Gpt-4 technical report.

Nicky Pochinkov. 2023. Llm modularity: The separability of capabilities in large language models. `https://www.lesswrong.com/posts/j84JhErNezMxyK4dH/llm-modularity-the-separability-of-capabilitie`

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv e-prints*.

Xavier Suau, Luca Zappella, and Nicholas Apostoloff. 2020. Finding experts in transformer models. *CoRR*, abs/2005.07647.

Zhengyan Zhang, Yankai Lin, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2021. Moefication: Conditional computation of transformer models for efficient inference. *CoRR*, abs/2110.01786.

A Compositional Explanations

Since we use tokens as inputs, the compositional explanation method we proposed, adapted from (Mu and Andreas, 2020), turns out to be very similar to looking at top-k activating tokens.

B Cluster Token Embeddings

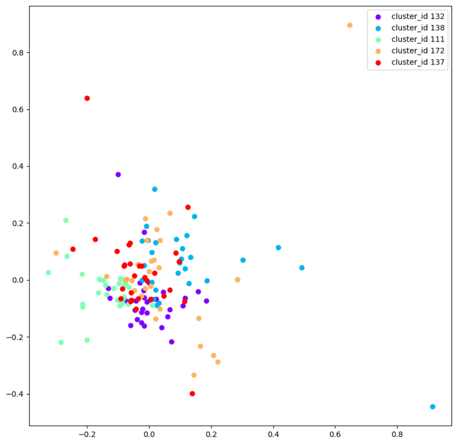


Figure 3: FastText token embeddings, projected with PCA, of the top-30 tokens in the filtered+smoothed method, with 50 clusters. The 5 clusters whose token embeddings are most closely scattered are plotted here.

C MLM Loss Visualization

D Visualization of Neuron Physical Positions

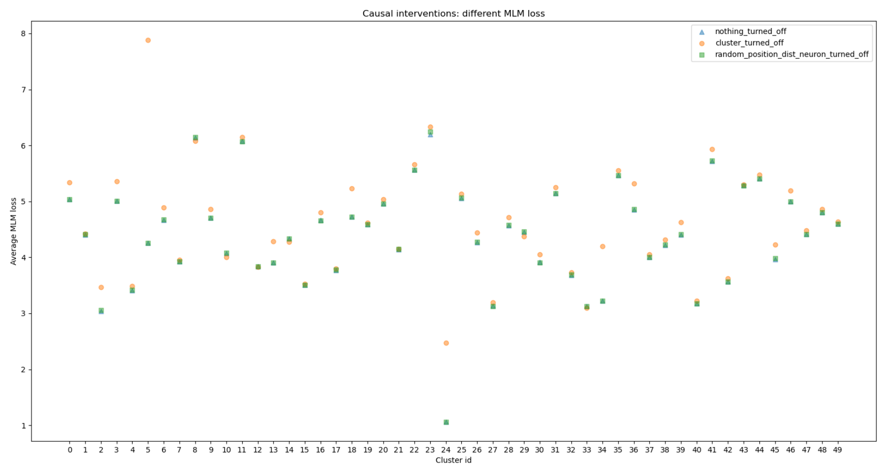
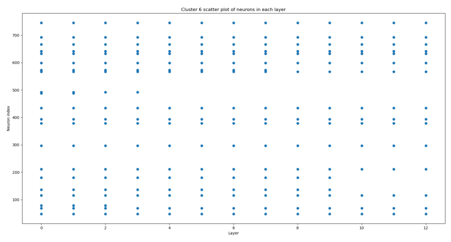
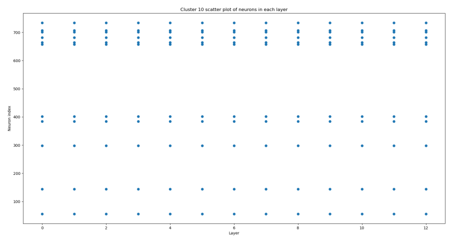


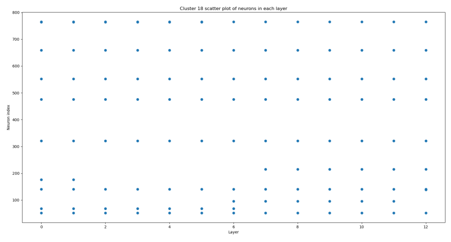
Figure 4: Visualization of the MLM loss for 50 clusters on the C4 dataset, comparing with and without the cluster, as well as the $random\_position$ baseline.



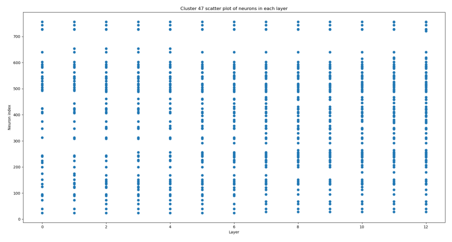
(a) Cluster 6



(b) Cluster 10



(c) Cluster 18



(d) Cluster 47

Figure 5: A few examples of distributions of neuron locations within each cluster. x-axis is layer and y-axis is the index position in the layer. Interestingly, neurons in each of our clusters are evenly distributed across layers and yet are at the same index positions in each layer.