Neuron-Level Knowledge Attribution in Large Language Models

Anonymous ACL submission

Abstract

Identifying important neurons for final predictions is essential for understanding the mechanisms of large language models. Due to computational constraints, current attribution techniques struggle to operate at neuron level. In this paper, we propose a static method for pinpointing significant neurons for different outputs. Compared to seven other methods, our approach demonstrates superior performance across three metrics. Additionally, since most static methods typically only identify "value neurons" directly contributing to the final prediction, we introduce a static method for identifying "query neurons" which activate these "value neurons". Finally, we apply our methods to analyze the localization of six distinct types of knowledge across both attention and feed-forward network (FFN) layers. Our method and analysis are helpful for understanding the mechanisms of knowledge storage and set the stage for future research in knowledge editing. We will release our data and code on github.

1 Introduction

Transformer-based large language models (LLMs) (Brown et al., 2020; Ouyang et al., 2022; Chowdhery et al., 2023) possess remarkable capabilities for storing factual knowledge, which is important for downstream tasks including question answering (Jiang et al., 2021) and reasoning (Rajani et al., 2019). While recent studies (Dai et al., 2021; Meng et al., 2022; Geva et al., 2023; Yu et al., 2023; Chen et al., 2024) have made significant progress in understanding knowledge localization and the information flow from inputs to predictions, it is still hard to identify exact parameters for knowledge storage in LLMs due to several reasons. Firstly, existing studies often depend on causal tracing (Pearl, 2001; Vig et al., 2020) and integrated gradients (Sundararajan et al., 2017) for knowledge attribution. However, many studies (Stolfo et al., 2023; Zhao et al., 2024; Wu et al., 2024a) point out that the computational complexity of forward and backward operations in these methods restricts their applicability to millions of neurons in LLMs, which are proved as fundamental units for storing knowledge (Geva et al., 2020; Dai et al., 2021; Geva et al., 2022; Nanda et al., 2023b). Secondly, while a few studies (Dar et al., 2022; Geva et al., 2022) have devised methods for analyzing neurons, they often lack comparisons with other methods. Therefore, how to identify important neurons in LLMs is still unclear. Lastly, existing methods typically concentrate on either attention or feed-forward network (FFN) module, often lacking evaluation of the other module. It is crucial to quantitatively compare the importance of both attention and FFN layers.

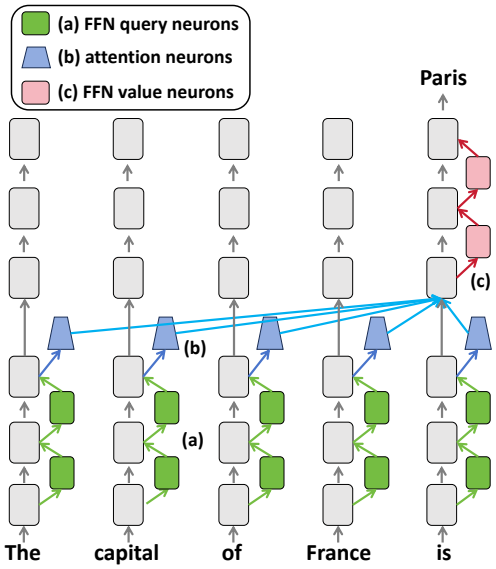

Figure 1: (a) Query neurons in shallow FFN layers. (b) Attention query/value neurons in attention heads. (c) Value neurons in deep FFN layers.

In this paper, we focus on neuron-level attribution methods. We analyze the distribution change caused by each neuron and discover that both the neuron's coefficient score and the final prediction's ranking, when projecting this neuron's subvalue into vocabulary space, play significant roles. Based on this finding, we employ log probability increase

as importance score, enabling the identification of neurons that contribute significantly to final predictions. Compared with seven other static methods, our proposed method achieves the best performance on three metrics. Furthermore, since the identified neurons directly contribute to the final predictions' probability, we also develop a static method to identify "query neurons" that aid in activating these "value neurons". Specifically, we calculate the inner products between the query neurons and value neurons as importance scores.

Based on our proposed methods, we analyze six types of knowledge in both attention and FFN layers, yielding numerous valuable insights (Figure 1): 1) Both attention and FFN layers can store knowledge, and all important neurons directly contribute to knowledge prediction are in deep layers. 2) In attention layers, knowledge with similar semantics (e.g. language, country, city) tends to be stored in the same heads. Knowledge with distinct semantics (e.g. country, color) is stored in different heads. 3) While numerous neurons contribute to the final prediction, intervening on a few value neurons (300) or query neurons (1000) can significantly influence the final prediction. 4) FFN value neurons are mainly activated by medium-deep attention value neurons, while these attention neurons are mainly activated by shallow/medium FFN query neurons.

Overall, our contributions are as follows:

a) We design a static method for neuron-level knowledge attribution in large language models. Compared with seven static methods, our method achieves the best performance under three metrics.

b) As the identified neurons usually directly contribute to the final predictions, we design a static method to identify the "query neurons" activating these "value neurons".

c) We analyze the localization of six types of knowledge in both attention and FFN layers. Our analysis is helpful for understanding the mechanisms of knowledge storage in language models.

2 Related Work

2.1 Attribution Methods for Transformers

Determining how to attribute the important parameters for final predictions is a crucial question. Gradient-based methods (Sundararajan et al., 2017; Kindermans et al., 2019; Miglani et al., 2020; Lundstrom et al., 2022) and causal tracing methods (Pearl, 2001; Vig et al., 2020; Meng et al., 2022; Goldowsky-Dill et al., 2023; Zhang and Nanda, 2023; Wu et al., 2024b; Hase et al., 2024) are widely utilized for this purpose. The core idea is calculating how much an internal module affects the final predictions, requiring multiple forward and/or backward operations (Wu et al., 2024a). Due to the computational overhead, these methods are usually applied on hidden states (Meng et al., 2022; Geva et al., 2023; Stolfo et al., 2023), rather than neurons. Another type of studies tend to require only one forward pass for each sentence, typically relying on saliency scores such as attention weights (Vig, 2019; Jaunet et al., 2021; Yeh et al., 2023; Wang et al., 2023; Li et al., 2023) and FFN neurons' coefficient scores (Geva et al., 2022; Lee et al., 2024). However, the validity of attributions is challenged by many studies (Serrano and Smith, 2019; Jain and Wallace, 2019; Wiegreffe and Pinter, 2019; Mohankumar et al., 2020; Ethayarajh and Jurafsky, 2021; Bai et al., 2021). Lack of evaluation methods results in an ongoing debate about the faithfulness of saliency score methods.

2.2 Mechanistic Interpretability

Mechanistic interpretility (Olah, 2022; Nanda et al., 2023a) aims to reverse engineer the circuits from inputs to the final prediction. An essential technology involves projecting internal vectors into the vocabulary space, where numerous studies have discovered interpretable results (Nostalgebraist, 2020; Geva et al., 2020, 2022; Dar et al., 2022; Pal et al., 2023). Most studies focus on analyzing the attention heads' roles in different cases and tasks (Elhage et al., 2021; Olsson et al., 2022; Wang et al., 2022; Hanna et al., 2023; Lieberum et al., 2023; Conmy et al., 2023; Gould et al., 2023). Also, superposition (Elhage et al., 2022; Nanda et al., 2023b; Gurnee et al., 2023) and dictionary learning (Bricken et al., 2023; He et al., 2024) are important for understanding neurons in transformers.

3 Methodology

In this section, we focus on static attribution methods to locate important neurons for specific predictions. We introduce the background in Section 3.1. Then we analyze the distribution change caused by neurons in Section 3.2. Based on the analysis, we introduce our proposed method for locating the "value neurons" that contribute to the final predictions directly in Section 3.3, and propose a static method to locate the "query neurons" that activate these "value neurons" in Section 3.4.

3.1 Background

First, we introduce the inference pass from inputs to the final prediction. Given an input sentence $X = [t_1, t_2, ..., t_T]$ with T tokens, the model generates the next token's probability distribution y over B tokens in vocabulary V. The embedding matrix $E \in \mathbb{R}^{B\times d}$ transforms each t_i at position i into a word embedding $h_i^0 \in \mathbb{R}^d$. Then the word embeddings are transformed by $L+1$ transformer layers ($0th - Lth$), each has a multi-head self-attention layer (MHSA) and a FFN layer. The layer output h_i^l (position i, layer l) is the sum of the layer input h_i^{l-1} (previous layer's output), the attention output A_i^l, and the FFN output F_i^l:

$$h_i^l = h_i^{l-1} + A_i^l + F_i^l \tag{1}$$

Finally, the last position's Lth layer output is used to compute the final probability distribution y by multiplying the unembedded matrix $E_u \in \mathbb{R}^{B\times d}$:

$$y = softmax(E_u\, h_T^L) \tag{2}$$

Specifically, the attention layer's output is computed by a weighted sum over H heads on T positions, and the FFN layer's output is computed by a nonlinear function σ on two linear transformations.

$$A_i^l = \sum_{j=1}^{H} ATTN_j^l(h_1^{l-1}, h_2^{l-1}..., h_T^{l-1}) \tag{3}$$

$$F_i^l = W_{fc2}^l \sigma(W_{fc1}^l(h_i^{l-1} + A_i^l)) \tag{4}$$

where $W_{fc1}^l \in \mathbb{R}^{N\times d}$ and $W_{fc2}^l \in \mathbb{R}^{d\times N}$ are two matrices. Geva et al. (2020) find FFN output can be represented as a weighted sum of FFN neurons:

$$F_i^l = \sum_{k=1}^{N} m_{i,k}^l fc2_k^l \tag{5}$$

$$m_{i,k}^l = \sigma(fc1_k^l \cdot (h_i^{l-1} + A_i^l)) \tag{6}$$

The FFN output F_i^l is computed by a weighted sum of $fc2$ vectors. $fc2_k^l$ is the kth column of W_{fc2}^l (named FFN subvalue), and its coefficient score $m_{i,k}^l$ is computed by non-linear σ on the inner product between the residual output $h_i^{l-1} + A_i^l$ and $fc1_k^l$ (named FFN subkey), the kth row of W_{fc1}^l. Similarly, the attention output A_i^l can be represented as a sum of head outputs, each being a weighted sum of value-output vectors on all positions:

$$A_i^l = \sum_{j=1}^{H}\sum_{p=1}^{T} \alpha_{i,j,p}^l W_{j,l}^o (W_{j,l}^v h_p^{l-1}) \tag{7}$$

$$\alpha_{i,j,p}^l = softmax(W_{j,l}^q h_i^{l-1} \cdot W_{j,l}^k h_p^{l-1}) \tag{8}$$

where $W_{j,l}^q, W_{j,l}^k, W_{j,l}^v, W_{j,l}^o \in \mathbb{R}^{d\times d/H}$ are the query, key, value and output matrices of the jth head in the lth layer. The query and key matrices compute the attention weight $\alpha_{i,j,p}^l$ on the pth position, then calculate the softmax function across all positions. The value and output matrices transform the pth position input vector into the pth value-output vector. Each head output is the weighted sum of value-output vectors on all positions.

As shown in Eq.1, Eq.5 and Eq.7, the final layer output can be represented as a direct addition of many vectors. Specifically, the final output can be regarded as a sum of $L \times (T \times H + N) + 1$ vectors. Moreover, the position value-output vector in Eq.7 can also be regarded as a weighted sum of output-matrix neurons, and the value-matrix neurons are the "subkeys" similar to Eq.6. If taking the output-matrix neurons as fundamental units, the final output is the sum of $L \times (d + N) + 1$ vectors.

3.2 Distribution Change Caused by Neurons

Since the final vector has essential information for predicting the final token, and it is computed by a direct sum of various neuron-level vectors, the relevant information for making the final prediction must be stored in one or many neurons. The final vector h_T^l can be regarded as the sum of one neuron v and another vector $x = h_T^l - v$. We consider the probability change $p(w|x+v) - p(w|x)$ caused by v for prediction token w. We aim to explore which components of v are significant for amplifying the probability change. This allows us to develop static methods for locating crucial neurons.

As the probability change is nonlinear, analyzing the exact contribution of neuron v is challenging. For a more concise analysis, we term the score $e_w \cdot x$ vector x's bs-value (before-softmax value) on token w, where e_w is the wth row of the unembedded matrix E_u. A token's bs-value directly corresponds to the probability of this token. Bs-values of all vocabulary tokens on vector x are:

$$bs(x) = [bs_1^x, bs_2^x, ..., bs_w^x, ..., bs_B^x] \tag{9}$$

For vector x, if bs_w^x is the largest among all the bs-values, the probability of word w will also be the highest. The probability of each token for x and $x + v$ can be computed by all the bs-values:

$$p(w|x) = \frac{exp(bs_w^x)}{exp(bs_1^x) + ... + exp(bs_B^x)} \tag{10}$$

$$p(w|x+v) = \frac{exp(bs_w^{x+v})}{exp(bs_1^{x+v}) + ... + exp(bs_B^{x+v})} \tag{11}$$

where bs-value bs_w^{x+v} is equal to $bs_w^x + bs_w^v$:

$$bs(x+v) = bs(x) + bs(v) \tag{12}$$

Although the probability change is nonlinear, the change on each token's bs-value is linear. In order to analyze which components of v is important, we design several $bs(x)$ and $bs(v)$ and compute the distribution change. Assume there are four tokens in vocabulary space, and $bs(x) = [1, 2, 3, 4]$. The probability distribution of x is $[0.03, 0.09, 0.24, 0.64]$. We design several v and compute the probability distribution of $p(x+v)$. The details are shown in Table 1.

$bs(v)$	$bs(x+v)$	$p(x+v)$
$[1, 1, 1, 3]$	$[2, 3, 4, 7]$	$[0.01, 0.02, 0.05, 0.93]$
$[3, 1, 1, 1]$	$[4, 3, 4, 5]$	$[0.20, 0.07, 0.20, 0.53]$
$[6, 4, 4, 4]$	$[7, 6, 7, 8]$	$[0.20, 0.07, 0.20, 0.53]$
$[6, 2, 2, 2]$	$[7, 4, 5, 6]$	$[0.64, 0.03, 0.09, 0.23]$
$-[6, 2, 2, 2]$	$[-5, 0, 1, 2]$	$[0.00, 0.09, 0.24, 0.67]$

Table 1: Probability distribution of $p(x+v)$.

Existing studies (Geva et al., 2022; Lee et al., 2024) state that $p(w|x+v) \propto exp(e_w \cdot v)$. However, based on the examples provided in Table 1, it appears that not only the bs-value of token w, but the bs-values of all the tokens affect the probability. For example, $bs(v) = [3, 1, 1, 1]$ and $[6, 4, 4, 4]$ result in the same distribution, although the bs-value of each token is enlarged.

An intuitive observation is that v aids in magnifying the token with the largest bs-value. For instance, $[1, 1, 1, 3]$ is conducive to increasing the probability of the last token, and $[3, 1, 1, 1]$ can amplify the probability of the first token. This observation may elucidate why many neurons exhibit human-interpretable concepts when projected into the vocabulary space. Given that the vocabulary size B is typically large (often exceeding 30,000), probabilities of tokens with the largest bs-values are likely to be augmented.

Another significant finding is that both the coefficient score and the neuron's bs-values play substantial roles. Compared with $[3, 1, 1, 1]$, $[6, 2, 2, 2]$ can can both amplify and diminish the probability change of $[3, 1, 1, 1]$. The probability increase of the first token is magnified, while the decrease in probability of the last token is more pronounced. When the coefficient score's sign is changed (e.g. $-[6, 2, 2, 2]$), the effect on the first token's probability changes from increasing to decreasing.

3.3 Importance Score for "Value Neurons"

Based on the analysis in Section 3.2, an intuitive importance score of a neuron mv is $|m| \times |1/rank(w)|$, where m is the coefficient score and $rank(w)$ denotes the ranking of the final token when projecting v into vocabulary space. Another intuitive importance score is calculating the probability $p(w|mv)$ on token w. If these scores are large, v will contain much information of w.

However, these methods have two potential problems. On one hand, they only consider the effect of v, overlooking the varying importance of v under different x conditions. On the other hand, we usually hope to analyze the importance of different modules' combination. Therefore, it is better that the importance score Imp satisfies $Imp(x+v) \approx Imp(x) + Imp(v)$.

To address these problems, we design log probability increase as importance score for both layer-level and neuron-level vectors. If v^l is a vector in lth attention layer, the importance score of v^l is:

$$Imp(v^l) = log(p(w|v^l + h^{l-1})) - log(p(w|h^{l-1})) \tag{13}$$

where the probability of each vector is computed by multiplying the vector with E_u (see Eq.2). If v^l is a vector in lth FFN layer, we compute the importance score by replacing h^{l-1} as $h^{l-1} + A^l$ in Eq.13. In Eq.13, v^l is not the only element controlling the importance score. Also, it is convenient for analyzing the combination of different modules.

3.4 Importance Score for "Query Neurons"

As discussed in Section 3.3, the proposed attribution methods can effectively identify the "value neurons" containing crucial information for the final prediction. However, in addition to these "value neurons", there exist "query neurons" that aid in activating these neurons, even if they may not directly contain information about w. In this section, we propose a static method to identify these "query neurons" based on Eq.1, Eq.5, and Eq.6. Since the $fc2$ vectors do not change, the coefficient scores are the only varying element in different cases. For each "value neuron", we can compute the inner product between its subkey (see Eq.6) and each

neuron/subvector within the residual output (see Eq.1). Despite the presence of a nonlinear function σ for computing the coefficient score, it usually does not affect the relative value between different neurons/subvectors. Therefore, if a "query" neuron/subvector exhibits a larger inner product with the subkey compared to another one, it is more helpful for activating the "value neuron".

4 Experiments

In this section, we compare our neuron-level attribution method with seven other methods in Section 4.1. Then we analyze the localization of six types of knowledge using our method in Section 4.2.

4.1 Comparison of Attribution Methods

We compare the proposed method in Eq.13 with seven other methods. For each sentence, we apply every method to identify top10 FFN neurons, and evaluate the attributed neurons using three metrics.

Dataset. To eliminate interference from specific prompts, we extract query-answer pairs with six types of answer tokens (color, month, number, language, capital, country) from TriviaQA (Joshi et al., 2017). To explore the mechanism of knowledge storage, we extract all the sentences where the correct token ranks within the top10 predictions and higher than other words in the same knowledge in GPT2-large (Radford et al., 2019) and Llama-7B (Touvron et al., 2023). There are 1,350 sentences for GPT2-large and 3,141 sentences for Llama-7B.

Models. To compare the differences between large and small models in terms of knowledge storage, we conduct experiments on Llama-7B and GPT2-large. Llama-7B consists of 32 layers, with each attention layer comprising 32 heads, each head containing 128 neurons and each FFN layer containing 11,008 neurons. GPT2-large has 36 layers with 20 heads per attention layer, 64 neurons per head, and 5,120 neurons per FFN layer.

Attribution methods. We compare our method with seven static methods. We use each method to attribute the FFN neurons with top10 scores for the correct knowledge token w. Similar to Eq.5, each neuron mv is the product of the coefficient score m and $fc2$ vector v^l. Here are the methods:
a) (proposed method) log probability increase: $log(p(w|mv^l + A^l + h^{l-1})) - log(p(w|A^l + h^{l-1}))$
b) log probability: $log(p(w|mv^l))$, which attributes the same neurons with $p(w|mv^l)$
c) probability increase: $p(w|mv^l + A^l + h^{l-1}) - p(w|A^l + h^{l-1})$
d) norm: $|v^l|$
e) coefficient score: $|m|$
f) ranking in vocabulary space: $1/rank(w)$
g) $|m| \times |v^l|$
h) $|m| \times 1/rank(w)$

Metrics. We devise three metrics to evaluate the attribution methods. After attributing the top10 FFN neurons by each method, we intervene on these neurons by setting the top10 neurons' parameters to zero. Subsequently, we rerun the model and compute the Mean Reciprocal Rank (MRR) score of the correct token w, the probability of w (prob), and the log probability of w (logp). An attribution method is considered superior when it exhibits greater decreases in these metrics.

	GPT2-large			Llama-7B		
	MRR	prob	logp	MRR	prob	logp
o)	0.361	7.1	-3.15	0.551	21.5	-2.24
a)	**0.201**	**3.4**	**-4.06**	**0.312**	**9.2**	**-3.91**
b)	0.214	3.6	-3.91	0.339	10.8	-3.35
c)	0.219	3.7	-3.92	0.345	10.0	-3.57
d)	0.363	7.1	-3.14	0.549	21.3	-2.25
e)	0.439	8.6	-3.10	0.529	22.9	-2.35
f)	0.306	5.8	-3.40	0.493	18.1	-2.49
g)	0.394	8.1	-3.06	0.523	22.6	-2.39
h)	0.232	4.0	-3.80	0.389	13.0	-3.06

Table 2: Results of attribution methods on two models.

Results and analysis. The results of the original model (first line) and eight attribution methods are shown in Table 1. In comparison with the other seven methods, our attribution method (second line) attributes more important neurons, resulting in the most significant reduction across all metrics in both GPT2 and Llama. Specifically, when only intervening ten FFN neurons, the probability of the correct knowledge token reduces from 7.1% to 3.4% in GPT2, and from 21.5% to 9.2% in Llama-7B. This indicates that there are several neurons storing much important information for knowledge storage, and our method can locate these neurons.

The attribution methods of norm v^l (d) and $m \times |v^l|$ (g) are not useful, which indicates the norm of neurons is not important for attribution. Using $|m| \times 1/rank(w)$ (h) has good results, which is better than $1/rank(w)$ (f). The ranking of tokens in vocabulary space for projected neurons is a good

saliency score, and the coefficient score can enlarge the distribution change, aligning our analysis in Section 3.2. Only using coefficient score (e) is not helpful for attribution. The role of coefficient score is to enhance the probability change caused by the neuron, but whether the neuron is useful for the selected token depends on the neuron itself. There are other tokens competing with the correct knowledge token, so the neurons with large coefficient scores may be related to these tokens.

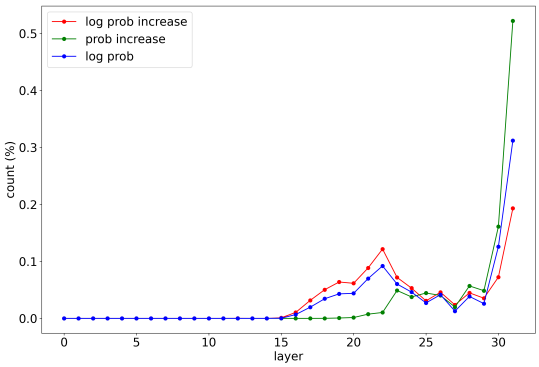

Figure 2: Neuron distribution on all layers in Llama-7B.

Compared to log probability (b), employing log probability increase (a) can attribute more important neurons. This aligns with the analysis in Section 3.2 and 3.3: not only neuron v, but also x affects $p(w|x+v)-p(w|x)$. Compared with probability increase (c), log probability increase achieves better results. We analyze the distribution of neurons across all layers in Llama attributed by log probability increase, log probability, and probability increase, as depicted in Figure 2. GPT2 has similar results, detailed in Appendix A. The neurons attributed by probability increase are on deepest layers ($23th - 31th$), while other two methods can attribute neurons among $17th$ to $31th$ layers.

To delve into the reason of this phenomenon, we analyze the difference of importance score when adding the same vector v on different x. As discussed in Eq.13, the importance score of v is computed by $log(p(w|x+v)) - log(p(w|x))$. Therefore, the importance score is related to the curve of $F(a) = log(p(w|a))$. To analyze this curve, we compute the final vector h_T^L and the $0th$ layer input vector h_T^0 on each sentence, and divide $h_T^L - h_T^0$ into 61 segments, where each segment is $Seg_S = h_T^0 + S(h_T^L - h_T^0)/60$ (S is the segment index from 0 to 60). Then we compute the probability $p(w|Seg_S)$ and log probability $log(p(w|Seg_S))$ at each segment index for every sentence. The average score on Llama-7B is shown in Figure 3.

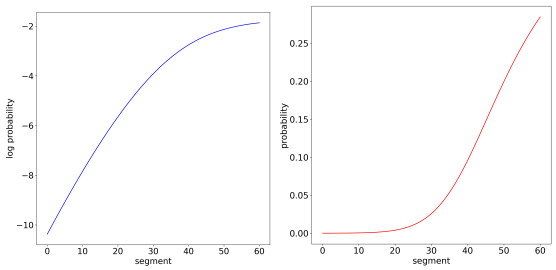

Figure 3: Curves of log probability increase (left) and probability increase (right) on Llama-7B.

The curve of log probability increase exhibits an approximately linear shape from 0 to 40 segments, while the curve of probability increase shows a linear trend from 40 to 60 segments. This observation elucidates the findings in Figure 2: employing probability increase is more inclined to attribute neurons in the deepest layers, whereas log probability increase tends to attribute neurons in medium-deep layers. Despite the slower slope of the log probability increase curve in very deep layers, it still effectively attributes neurons in very deep layers (as depicted in Figure 2). This maybe because neurons in very deep layers contain substantial information, and even when the importance score decreases, it remains relatively large. In later sections, we use log probability increase as importance score for exploration, as this method can identify the important neurons in both medium-deep layers and very deep layers, and its experimental results are the best. Nevertheless, reproducing the importance of the deepest layers may be a prospective avenue for developing improved attribution methods.

4.2 Exploration on Different Knowledge

We take log probability increase as importance score, and analyze six types of knowledge: language (lang), color (col), number (num), capital (capi), country (cnty), and month (mon). We evaluate the knowledge storage in attention and FFN layers at layer-level, head-level, and neuron-level.

	lang	col	num	capi	cnty	mon
G-A	7.46	4.10	2.75	6.61	6.62	4.82
G-F	0.63	3.51	4.70	1.89	1.67	2.81
L-A	6.28	3.81	2.22	5.03	6.42	5.10
L-F	1.74	4.02	5.60	4.30	2.90	3.05

Table 3: Contribution of attention and FFN layers.

Layer-level knowledge storage. We compute the sum of importance score of each attention and

FFN layer in GPT2 (G-A, G-F) and Llama (L-A, L-F), shown in Table 3. Also, we calculate the top10 layer for each knowledge in Table 4, where a_l and f_l means lth attention and FFN layer.

Both attention and FFN layers have ability to store knowledge, and all the top10 layers are in deep layers. Information with analogous semantics (e.g., language, capital, country) tends to be stored within similar layers/modules. For instance, a_{26}, a_{30}, a_{28}, and a_{22} in GPT2 ranks top for language, capital and country, and a_{23} in Llama-7B ranks the first for these knowledge. Data with dissimilar semantics (e.g., language, color, month) typically resides in distinct layers/modules.

	top10 important layers
lang	$a_{26}, a_{30}, a_{32}, a_{22}, a_{31}, a_{28}, a_{23}, a_{27}, a_{19}, a_{23}$
col	$a_{32}, f_{32}, a_{33}, f_{29}, f_{31}, a_{31}, a_{26}, f_{33}, f_{28}, a_{22}$
num	$f_{29}, f_{23}, f_{27}, f_{30}, f_{31}, f_{26}, f_{32}, a_{23}, a_{22}, f_{28}$
capi	$a_{26}, a_{28}, a_{30}, a_{25}, a_{22}, f_{26}, f_{28}, a_{19}, f_{27}, f_{30}$
cnty	$a_{26}, a_{30}, a_{28}, a_{22}, f_{29}, a_{31}, f_{26}, a_{32}, a_{25}, a_{19}$
mon	$a_{27}, a_{26}, f_{26}, a_{25}, f_{30}, a_{28}, a_{24}, a_{22}, a_{30}, f_{27}$
lang	$a_{23}, a_{21}, f_{21}, a_{19}, a_{18}, a_{31}, a_{25}, a_{16}, f_{20}, f_{19}$
col	$f_{29}, a_{20}, f_{22}, f_{20}, a_{19}, a_{28}, a_{16}, a_{29}, a_{18}, f_{28}$
num	$f_{31}, f_{26}, f_{29}, f_{27}, a_{26}, f_{23}, f_{24}, a_{28}, f_{17}, f_{30}$
capi	$a_{23}, f_{21}, f_{22}, a_{18}, a_{25}, a_{21}, f_{19}, f_{20}, a_{16}, f_{24}$
cnty	$a_{23}, a_{21}, a_{25}, f_{22}, a_{18}, a_{19}, a_{16}, f_{21}, f_{31}, a_{31}$
mon	$a_{21}, a_{19}, f_{19}, a_{16}, f_{31}, a_{23}, a_{28}, f_{30}, f_{17}, f_{18}$

Table 4: Top10 important layers in GPT2 (first block) and Llama (second block).

Head-level knowledge storage. We compute the importance score of each head (detailed in Appendix B) and find that many heads have ability to store similar knowledge. In GPT2, a_{30}^{6} (30th layer 6th head), $a_{26}^{17}, a_{32}^{11}, a_{25}^{13}$ and a_{22}^{17} rank top8 for language, capital and country. Similarly, $a_{23}^{12}, a_{19}^{31}, a_{31}^{25}$, and a_{25}^{25} rank top5 for these knowledge in Llama.

To evaluate how much knowledge the top heads store, we intervene the top 1% heads (top7 in GPT2 and top10 in Llama) by setting the heads' parameters to zero. Intervening each knowledge's heads result in a MRR/probability decrease of 44.5%/53.3% in GPT2, and 32.8%/48.2% in Llama (shown in Appendix B). But semantic-unrelated knowledge only reduce 7.1%/9.5% in GPT2 and 3.8%/8.7% in Llama. Therefore, the identified "knowledge heads" contain much semantic-related knowledge.

Neuron-level knowledge storage. For attention and FFN layers in Llama, we compute the sum of importance score for all neurons, all positive neurons (score larger than 0), top100 neurons, and top200 neurons, as illustrated in Table 5. Similar results of GPT2 is shown in Appendix C.

	lang	col	num	capi	cnty	mon
(attn) all	6.7	4.0	2.4	5.5	6.9	5.4
positive	30.5	32.0	24.8	29.4	30.5	29.2
top100	3.5	2.8	2.0	3.0	3.6	2.8
top200	5.0	4.1	3.0	4.4	5.0	4.1
(FFN) all	2.5	4.9	6.5	5.1	3.6	2.8
positive	77.4	90.6	71.6	77.6	69.8	69.1
top100	6.4	6.1	6.6	6.3	5.5	7.0
top200	8.2	8.0	8.5	8.0	7.0	8.5

Table 5: Imporatnce of top neurons in attention (first block) and FFN (second block) layers in Llama-7B.

In both models, the sum score of top200 neurons in attention layers and top100 neurons in FFN layers are similar to that of all neurons. Additionally, we intervene the top neurons to evaluate how much final predictions are affected, detailed in Appendix C. When intervening the top200 attention neurons and top100 FFN neurons for each sentence, the MRR and probability decreases 96.3%/99.2% in GPT2, and 96.9%/99.6% in Llama. In comparison, randomly intervening the same number of neurons only result a decrease of 0.22%/0.14%. Hence, even though there are many neurons contribute to the final prediction, intervening a few neurons (300) affects the final prediction much. This conclusion holds significance for future studies delving into neuron-level knowledge editing.

	top10 query layers for FFN neurons
lang	$a_{26}, a_{22}, a_{19}, f_{26}, a_{17}, f_{25}, f_{27}, f_{23}, a_{25}, a_{23}$
col	$f_{26}, f_{29}, a_{26}, f_{28}, f_{25}, a_{22}, f_{27}, a_{17}, a_{24}, f_{23}$
num	$f_{26}, f_{23}, f_{27}, a_{22}, f_{25}, a_{17}, f_{19}, f_{21}, a_{23}, f_{0}$
capi	$a_{26}, a_{22}, a_{19}, a_{17}, f_{25}, a_{23}, a_{18}, f_{26}, f_{21}, a_{28}$
cnty	$a_{26}, a_{22}, a_{17}, a_{19}, a_{23}, f_{18}, a_{20}, a_{18}, f_{21}, f_{25}$
mon	$a_{17}, a_{22}, f_{23}, a_{26}, f_{26}, f_{24}, a_{19}, a_{20}, f_{20}, f_{21}$
lang	$f_{21}, a_{16}, a_{19}, f_{18}, a_{18}, a_{21}, a_{17}, f_{30}, f_{19}, a_{14}$
col	$f_{20}, f_{21}, a_{15}, a_{17}, a_{18}, a_{20}, f_{19}, f_{22}, f_{17}, a_{16}$
num	$f_{19}, f_{21}, f_{22}, f_{16}, a_{18}, a_{22}, a_{24}, f_{14}, a_{12}, a_{25}$
capi	$a_{18}, a_{16}, f_{23}, a_{19}, f_{17}, a_{14}, f_{22}, a_{21}, f_{26}, f_{19}$
cnty	$a_{16}, a_{18}, a_{21}, a_{19}, f_{21}, a_{14}, f_{19}, a_{17}, a_{31}, f_{20}$
mon	$a_{16}, a_{19}, f_{18}, a_{21}, a_{17}, a_{14}, f_{29}, f_{19}, f_{17}, a_{18}$

Table 6: Top10 query layers for top100 FFN neurons in GPT2 (first block) and Llama (second block).

Important query layers for FFN value neurons. The "value" FFN neurons are activated by last position's residual stream. We evaluate which layers

activate the top100 FFN neurons, shown in Table 6. The medium-deep attention layers play large rules. Most of them also contribute to final predictions (e.g. a_{19}, a_{22}, a_{26} in GPT2 and $a_{16}, a_{18}, a_{19}, a_{21}$ in Llama for country/capital/language). Therefore, the medium-deep attention neurons are very important, working as both "value" and "query".

	lang	col	num	capi	cnty	mon
G	91/96	95/97	96/96	98/99	89/94	83/88
L	78/92	85/93	90/96	84/93	91/98	94/98

Table 7: MRR/probability decrease (%) when intervening 1,000 query neurons in GPT2 (G) and Llama (L).

Important query neurons for attention value neurons. We compute the important query layers activating the top200 "value" attention neurons. finding the shallow and medium FFN layers play main roles (detailed in Appendix D). To identify the important query FFN neurons, we weighted sum the inner product between each attention neuron's subkey and each FFN neuron on every position's residual stream, as query FFN neurons' scores. When intervening top1000 shallow neurons for each sentence, both MRR and probability drops very much (92%/95% in GPT2 and 87%/95% in Llama), shown in Table 7. In comparison, randomly intervening 1,000 neurons only result in a decrease of 0.8%/1.1%. Hence, our method can locate the important query neurons in these layers.

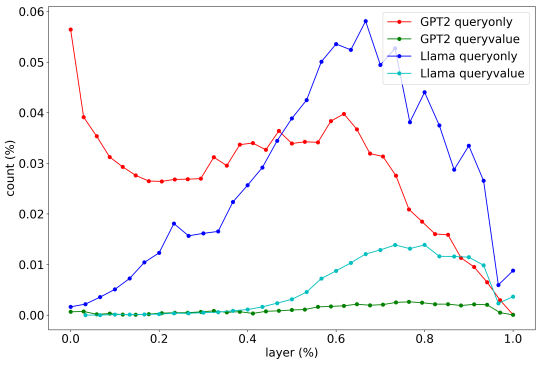

Figure 4: Query neuron distribution in GPT2 and Llama.

Then we count the number of queryvalue (both in top1000 query and top1000 value) and queryonly (only in top1000 query) FFN neurons, shown in Figure 4. In both models, the number of queryonly neurons, which is much larger than that of queryvalue neurons, starts to drop at 60% layer. This observation indicates that the shallow and medium FFN neurons are important for activating the attention "value neurons". A difference is that the very shallow FFN layers play large roles in GPT2, and we defer this exploration to future research. Overall, our analysis learns the information flow at neuron level: features in shallow/medium FFN neurons are extracted, then activate the deep attention and FFN neurons related to final predictions.

	lang	col	num	capi	cnty	mon
G-query	5	17	104	51	26	81
G-value	48	57	121	71	64	137
L-query	1	9	39	1	1	44
L-value	13	23	84	18	21	95

Table 8: Shared neurons in GPT2 (G) and Llama (L).

Shared Value and query neurons in each knowledge. We compute how many "shared" query neurons and value neurons rank top300 in more than 50% sentences in each knowledge, shown in Table 8. On average, there are 27.6% shared value neurons in GPT2 and 14.1% in Llama. Query neurons, with 15.7% shared neurons in GPT2 and 5.2% in Llama, exhibit a more dispersed distribution than value neurons. To explore the neurons' interpretability, we project them into vocabulary space. We find most value neurons (first block in Table 9) are related to predicted tokens. However, we do not observe much interpretability in query neurons. We only find a few query neurons (second block in Table 9) related to the final words. Hence, to explore the interpretability of query neurons may be a valuable direction in future works.

neuron	top10 tokens in vocabulary space
f_{29}-3771 (GPT2,v)	Chile,Nicaragua,Finland,Ireland,Belarus, Norway,Slovakia,Latvia,Australia
a_{23}^{12}-70 (Llama,v)	German,Greek,Netherlands,Dutch, Germany,Greece,Holland,Norwegian
f_0-2947 (GPT2,q)	Lion, Bull, Liver, riot, Gladiator, **Red**, uct, Ant, Les
f_7-8744 (Llama,q)	**Belgium**,Ireland,Vienna,Czech,Kas, Kansas,Netherlands,Iowa,wings,Spanish

Table 9: Interpretable neurons in vocabulary space.

5 Conclusion

In this study, we propose a method based on log probability increase to identify the important "value neurons". We also develop a method based on inner products to locate the "query neurons" activating these "value neurons". Our method and analysis on six types of knowledge are helpful for exploring and understanding the mechanism of LLMs.

6 Limitations

The first limitation of our study is that it focuses on six specific types of knowledge, while other types of knowledge are also important. Secondly, our experiments are conducted using GPT2-large and Llama-7B models. It is essential to compare the similarities and differences in knowledge storage across different models. Lastly, our study employs static methods for neuron-level knowledge attribution. Although our experiments demonstrate the correctness and robustness of our designed method, it is also important to compare static methods with other attribution methods, such as causal mediation analysis and gradient-based methods. We plan to explore these areas in future work.

A potential risk of our work is that people can utilize our method to identify important neurons and edit them to change the models' outputs. For instance, if they identify the toxicity neurons and gender bias neurons and increase these neurons' coefficient scores, the model will be more likely to generate toxicity and gender bias words. But this potential risk depends on how people utilize our method. Our method can be utilized for reducing hallucinations, toxicity, and bias in LLMs by identifying and intervening/editing these neurons.

References

Bing Bai, Jian Liang, Guanhua Zhang, Hao Li, Kun Bai, and Fei Wang. 2021. Why attentions may not be interpretable? In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 25–34.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Zac Hatfield-Dodds, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E Burke, Tristan Hume, Shan Carter, Tom Henighan, and Christopher Olah. 2023. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2023/monosemantic-features/index.html.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Yuheng Chen, Pengfei Cao, Yubo Chen, Kang Liu, and Jun Zhao. 2024. Journey to the center of the knowledge neurons: Discoveries of language-independent knowledge neurons and degenerate knowledge neurons. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17817–17825.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

Arthur Conmy, Augustine Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. 2023. Towards automated circuit discovery for mechanistic interpretability. *Advances in Neural Information Processing Systems*, 36:16318–16352.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2021. Knowledge neurons in pretrained transformers. *arXiv preprint arXiv:2104.08696*.

Guy Dar, Mor Geva, Ankit Gupta, and Jonathan Berant. 2022. Analyzing transformers in embedding space. *arXiv preprint arXiv:2209.02535*.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, et al. 2022. Toy models of superposition. *arXiv preprint arXiv:2209.10652*.

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.

Kawin Ethayarajh and Dan Jurafsky. 2021. Attention flows are shapley value explanations. *arXiv preprint arXiv:2105.14652*.

Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. 2023. Dissecting recall of factual associations in auto-regressive language models. *arXiv preprint arXiv:2304.14767*.

Mor Geva, Avi Caciularu, Kevin Ro Wang, and Yoav Goldberg. 2022. Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space. *arXiv preprint arXiv:2203.14680*.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2020. Transformer feed-forward layers are key-value memories. *arXiv preprint arXiv:2012.14913*.

Nicholas Goldowsky-Dill, Chris MacLeod, Lucas Sato, and Aryaman Arora. 2023. Localizing model behavior with path patching. *arXiv preprint arXiv:2304.05969*.

Rhys Gould, Euan Ong, George Ogden, and Arthur Conmy. 2023. Successor heads: Recurring, interpretable attention heads in the wild. *arXiv preprint arXiv:2312.09230*.

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 preprint arXiv:2305.01610*.

Michael Hanna, Ollie Liu, and Alexandre Variengien. 2023. How does gpt-2 compute greater-than?: Interpreting mathematical abilities in a pre-trained language model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Peter Hase, Mohit Bansal, Been Kim, and Asma Ghandeharioun. 2024. Does localization inform editing? surprising differences in causality-based localization vs. knowledge editing in language models. *Advances in Neural Information Processing Systems*, 36.

Zhengfu He, Xuyang Ge, Qiong Tang, Tianxiang Sun, Qinyuan Cheng, and Xipeng Qiu. 2024. Dictionary learning improves patch-free circuit discovery in mechanistic interpretability: A case study on othello-gpt. *arXiv preprint arXiv:2402.12201*.

Sarthak Jain and Byron C Wallace. 2019. Attention is not explanation. *arXiv preprint arXiv:1902.10186*.

Theo Jaunet, Corentin Kervadec, Romain Vuillemot, Grigory Antipov, Moez Baccouche, and Christian Wolf. 2021. Visqa: X-raying vision and language reasoning in transformers. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):976–986.

Zhengbao Jiang, Jun Araki, Haibo Ding, and Graham Neubig. 2021. How can we know when language models know? on the calibration of language models for question answering. *Transactions of the Association for Computational Linguistics*, 9:962–977.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.

Pieter-Jan Kindermans, Sara Hooker, Julius Adebayo, Maximilian Alber, Kristof T Schütt, Sven Dähne, Dumitru Erhan, and Been Kim. 2019. The (un) reliability of saliency methods. *Explainable AI: Interpreting, explaining and visualizing deep learning*, pages 267–280.

Andrew Lee, Xiaoyan Bai, Itamar Pres, Martin Wattenberg, Jonathan K Kummerfeld, and Rada Mihalcea. 2024. A mechanistic understanding of alignment algorithms: A case study on dpo and toxicity. *arXiv preprint arXiv:2401.01967*.

Zongxia Li, Paiheng Xu, Fuxiao Liu, and Hyemi Song. 2023. Towards understanding in-context learning with contrastive demonstrations and saliency maps. *arXiv preprint arXiv:2307.05052*.

Tom Lieberum, Matthew Rahtz, János Kramár, Geoffrey Irving, Rohin Shah, and Vladimir Mikulik. 2023. Does circuit analysis interpretability scale? evidence from multiple choice capabilities in chinchilla. *arXiv preprint arXiv:2307.09458*.

Daniel D Lundstrom, Tianjian Huang, and Meisam Razaviyayn. 2022. A rigorous study of integrated gradients method and extensions to internal neuron attributions. In *International Conference on Machine Learning*, pages 14485–14508. PMLR.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*, 35:17359–17372.

Vivek Miglani, Narine Kokhlikyan, Bilal Alsallakh, Miguel Martin, and Orion Reblitz-Richardson. 2020. Investigating saturation effects in integrated gradients. *arXiv preprint arXiv:2010.12697*.

Akash Kumar Mohankumar, Preksha Nema, Sharan Narasimhan, Mitesh M Khapra, Balaji Vasan Srinivasan, and Balaraman Ravindran. 2020. Towards transparent and explainable attention models. *arXiv preprint arXiv:2004.14243*.

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. 2023a. Progress measures for grokking via mechanistic interpretability. *arXiv preprint arXiv:2301.05217*.

Neel Nanda, Senthooran Rajamanoharan, Janos Kramar, and Rohin Shah. 2023b. Fact finding: Attempting to reverse-engineer factual recall on the neuron level. In *Alignment Forum*, page 6.

Nostalgebraist. 2020. Interpreting gpt: the logit lens.

Chris Olah. 2022. Mechanistic interpretability, variables, and the importance of interpretable bases. In *Transformer Circuits Thread*.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. 2022. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Koyena Pal, Jiuding Sun, Andrew Yuan, Byron C Wallace, and David Bau. 2023. Future lens: Anticipating subsequent tokens from a single hidden state. *arXiv preprint arXiv:2311.04897*.

Judea Pearl. 2001. Direct and indirect effects. *Probabilistic and Causal Inference: The Works of Judea Pearl*, page 373.

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.

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Explain yourself! leveraging language models for commonsense reasoning. *arXiv preprint arXiv:1906.02361*.

Sofia Serrano and Noah A Smith. 2019. Is attention interpretable? *arXiv preprint arXiv:1906.03731*.

Alessandro Stolfo, Yonatan Belinkov, and Mrinmaya Sachan. 2023. A mechanistic interpretation of arithmetic reasoning in language models using causal mediation analysis. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7035–7052.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Jesse Vig. 2019. Bertviz: A tool for visualizing multihead self-attention in the bert model. In *ICLR workshop: Debugging machine learning models*, volume 3.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. 2020. Investigating gender bias in language models using causal mediation analysis. *Advances in neural information processing systems*, 33:12388–12401.

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 preprint arXiv:2211.00593*.

Lean Wang, Lei Li, Damai Dai, Deli Chen, Hao Zhou, Fandong Meng, Jie Zhou, and Xu Sun. 2023. Label words are anchors: An information flow perspective for understanding in-context learning. *arXiv preprint arXiv:2305.14160*.

Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. *arXiv preprint arXiv:1908.04626*.

Xuansheng Wu, Haiyan Zhao, Yaochen Zhu, Yucheng Shi, Fan Yang, Tianming Liu, Xiaoming Zhai, Wenlin Yao, Jundong Li, Mengnan Du, et al. 2024a. Usable xai: 10 strategies towards exploiting explainability in the llm era. *arXiv preprint arXiv:2403.08946*.

Zhengxuan Wu, Atticus Geiger, Thomas Icard, Christopher Potts, and Noah Goodman. 2024b. Interpretability at scale: Identifying causal mechanisms in alpaca. *Advances in Neural Information Processing Systems*, 36.

Catherine Yeh, Yida Chen, Aoyu Wu, Cynthia Chen, Fernanda Viégas, and Martin Wattenberg. 2023. Attentionviz: A global view of transformer attention. *IEEE Transactions on Visualization and Computer Graphics*.

Qinan Yu, Jack Merullo, and Ellie Pavlick. 2023. Characterizing mechanisms for factual recall in language models. *arXiv preprint arXiv:2310.15910*.

Fred Zhang and Neel Nanda. 2023. Towards best practices of activation patching in language models: Metrics and methods. *arXiv preprint arXiv:2309.16042*.

Haiyan Zhao, Hanjie Chen, Fan Yang, Ninghao Liu, Huiqi Deng, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, and Mengnan Du. 2024. Explainability for large language models: A survey. *ACM Transactions on Intelligent Systems and Technology*, 15(2):1–38.

A Neuron Distribution in GPT2

The neuron distribution of GPT2-large is similar to Llama-7B, which is shown in Figure 5. Also, the curve of importance score in GPT2-large is similar to that in Llama-7B, illustrated in Figure 6.

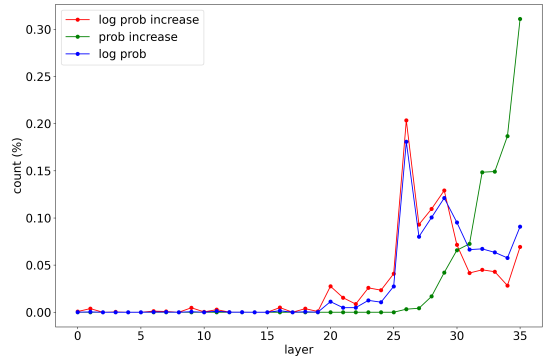

Figure 5: Neuron distribution on all layers in GPT2.

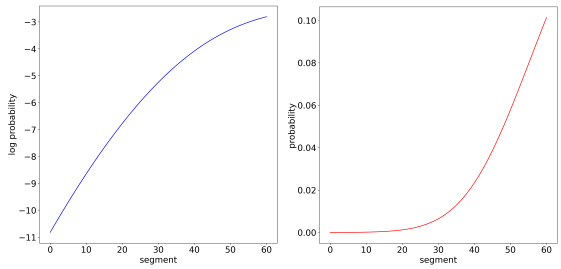

Figure 6: Curves of log probability increase (left) and probability increase (right) on GPT2.

B Head-Level Storage in GPT2/Llama

The top10 heads are shown in Table 10, where a_l^j is short for the jth head in lth attention layer. Knowledge with similar semantics is stored in the same heads (e.g. a_{30}^6 in GPT2 and a_{23}^{12} in Llama).

type	top10 heads
lang	$a_{30}^6, a_{26}^{17}, a_{26}^7, a_{32}^{11}, a_{19}^0, a_{31}^9, a_{25}^{13}, a_{22}^{17}, a_{28}^{13}, a_{29}^2$
col	$a_{33}^5, a_{34}^1, a_{26}^7, a_{24}^{19}, a_{23}^{18}, a_{32}^{13}, a_{30}^1, a_{22}^8, a_{32}^{14}, a_{28}^2$
num	$a_{22}^{18}, a_{17}^3, a_{23}^8, a_{19}^2, a_{30}^3, a_{25}^{19}, a_{20}^3, a_{30}^0, a_{12}^2, a_{25}^3$
capi	$a_{26}^7, a_{30}^6, a_{26}^{17}, a_{22}^{17}, a_{25}^{13}, a_{28}^{13}, a_{19}^0, a_{19}^{10}, a_{29}^2, a_{32}^{11}$
cnty	$a_{26}^7, a_{30}^6, a_{22}^{17}, a_{28}^{13}, a_{26}^{17}, a_{32}^{11}, a_{19}^0, a_{25}^{13}, a_{31}^9, a_{19}^{10}$
mon	$a_{27}^2, a_{26}^7, a_{25}^{11}, a_{19}^{10}, a_{30}^2, a_{28}^4, a_{23}^{18}, a_{17}^{17}, a_{33}^1, a_{17}^3$
lang	$a_{23}^{12}, a_{19}^{31}, a_{25}^{25}, a_{25}^{25}, a_{16}^5, a_{18}^1, a_{21}^9, a_{29}^{22}, a_{21}^{17}, a_{18}^{23}$
col	$a_{29}^{22}, a_{28}^{19}, a_{20}^{27}, a_{16}^{15}, a_{17}^{27}, a_{28}^{21}, a_{25}^{14}, a_{18}^{28}, a_{24}^1, a_{14}^3$
num	$a_{28}^{19}, a_{26}^{24}, a_{23}^{10}, a_{30}^{13}, a_{21}^{29}, a_{13}^{24}, a_{18}^{24}, a_{29}^{22}, a_{17}^{23}, a_{19}^1$
capi	$a_{23}^{12}, a_{29}^{22}, a_{25}^{25}, a_{31}^{25}, a_{19}^{31}, a_{18}^1, a_{16}^{15}, a_{16}^5, a_{21}^9, a_{18}^{23}$
cnty	$a_{23}^{12}, a_{19}^{31}, a_{25}^{25}, a_{21}^9, a_{31}^{25}, a_{16}^{15}, a_{18}^1, a_{16}^5, a_{29}^{22}, a_{28}^{19}$
mon	$a_{21}^{10}, a_{16}^0, a_{21}^{22}, a_{23}^{18}, a_{28}^{16}, a_{19}^{20}, a_{31}^6, a_{19}^1, a_{14}^3, a_{20}^{13}$

Table 10: Top10 important heads in GPT2 (first block) and Llama (second block).

The MRR decrease (%)/probability decrease (%) when intervening the top 1% heads for each knowledge is shown in Table 11. When intervening the top 1% heads for each knowledge, similar knowledge (language, capital and country) is affected a lot, while other knowledge (month, color, number) is not affected much.

	lang	capi	coun	mon	col	num
lang	**44/51**	**33/52**	**38/56**	3/0	6/5	1/2
capi	**32/38**	**42/53**	**39/54**	15/11	18/12	0/1
coun	**39/44**	**40/54**	**44/60**	3/0	10/5	2/3
mon	19/24	14/19	13/19	**55/63**	24/23	5/4
col	6/6	1/0	2/4	13/12	**49/59**	5/7
num	11/14	2/10	7/11	17/21	19/16	**33/34**
lang	**24/42**	**17/35**	**13/33**	0/0	6/15	1/1
capi	**38/58**	**28/50**	**22/53**	1/0	7/16	0/0
coun	**42/61**	**31/54**	**28/58**	0/0	10/21	2/6
mon	7/14	4/11	7/13	**51/66**	8/13	3/8
col	3/12	6/16	6/14	13/25	**33/42**	1/10
num	0/0	1/4	1/2	2/9	3/13	**33/31**

Table 11: MRR decrease (%)/probability decrease (%) in GPT2 (first block) and Llama (second block) when intervening top 1% heads for different knowledge.

C Neuron-Level Storage in GPT2/Llama

The sum score of top neurons and all neurons in GPT2 are shown in Table 12. The sum importance score of top200 attention neurons and top100 FFN neurons are similar to those of all neurons.

	lang	col	num	capi	cnty	mon
all	7.3	3.6	2.3	6.7	6.6	4.3
positive	28.8	24.7	18.2	27.4	27.5	23.3
top100	4.0	2.7	1.6	3.7	3.5	2.5
top200	5.7	4.0	2.5	5.3	5.1	3.7
all	0.0	2.5	4.0	1.9	1.4	1.9
positive	70.1	72.0	62.3	69.4	69.1	66.6
top100	4.2	4.3	4.1	4.6	4.3	4.7
top200	6.0	6.1	5.7	6.3	5.9	6.4

Table 12: Imporatnce of top neurons in attention (first block) and FFN (second block) layers in GPT2.

The MRR decrease (%) / probability decrease (%) when intervening the top 200 attention neurons and top100 FFN neurons are shown in Table 13. The MRR score and probability score decreases around 91.1%/98.7% in GPT2, and 88.4%/97.1% in Llama. Therefore, our method can identify the important "value neurons" in both attention and FFN layers.

	lang	col	num	capi	cnty	mon
G	96/99	97/99	96/98	96/99	97/99	96/99
L	97/99	98/99	96/99	97/99	97/99	97/99

Table 13: MRR decrease (%) and probability decrease (%) when intervening the top200 attention neurons and top100 FFN neurons in GPT (G) and Llama (L).

D Important Query Layers for Attention Neurons in GPT2/Llama

We evaluate which layers have large inner product with top200 attention neurons, shown in Table 14. For every knowledge, the shallow and medium FFN layers play larger roles than attention layers.

	top10 query layers for attention neurons
lang	$f_0, f_1, a_0, f_2, f_{19}, f_{20}, f_3, f_{17}, f_{18}, f_{21}$
col	$f_0, f_1, f_2, f_{23}, f_{20}, f_{21}, f_{22}, f_{24}, a_0, f_3$
num	$f_0, f_{18}, f_1, f_{19}, f_{22}, f_{16}, f_{21}, f_2, f_{12}, f_{20}$
capi	$f_0, f_1, a_0, f_2, f_3, f_{20}, f_5, f_4, f_{19}, f_{17}$
cnty	$f_0, f_1, f_{19}, a_0, f_{18}, f_2, f_3, f_{21}, f_{20}, f_{17}$
mon	$f_0, f_1, f_{19}, f_2, f_9, f_{22}, f_{10}, f_{21}, a_0, f_{18}$
lang	$f_{20}, f_{19}, a_{16}, f_{16}, f_{15}, f_{18}, f_{14}, f_{21}, f_{12}, f_{21}$
col	$f_{15}, f_{18}, f_{20}, f_{16}, f_{19}, f_{13}, f_{17}, f_{22}, f_{24}, f_{14}$
num	$f_{24}, f_{17}, f_{19}, f_{23}, f_{22}, f_{20}, f_{18}, f_2, f_{25}, f_{21}$
capi	$f_{20}, f_{24}, f_{22}, f_{23}, f_{19}, a_{16}, a_{23}, f_{28}, f_{18}, f_{25}$
cnty	$f_{18}, f_{21}, f_{19}, a_{18}, f_{22}, a_{14}, a_{16}, f_{17}, a_{21}, a_{19}$
mon	$f_{14}, a_{16}, f_{19}, a_{19}, f_{18}, f_{20}, f_{13}, f_{17}, f_{15}, f_{22}$

Table 14: Top10 query layers for top200 attention neurons in GPT2 (first block) and Llama (second block).