Weight-based Analysis of Detokenization in Language Models: Understanding the First Stage of Inference Without Inference

Go Kamoda[1] **Benjamin Heinzerling**[2, 1] **Tatsuro Inaba**[3] **Keito Kudo**[1]
Keisuke Sakaguchi[1, 2] **Kentaro Inui**[4, 1, 2]

[1]Tohoku University [2]RIKEN [3]Kyoto University [4]MBZUAI

Correspondence: go.kamoda@dc.tohoku.ac.jp

Abstract

According to the stages-of-inference hypothesis, early layers of language models map their subword-tokenized input, which does not necessarily correspond to a linguistically meaningful segmentation, to more meaningful representations that form the model's "inner vocabulary". Prior analysis of this *detokenization* stage has predominantly relied on probing and interventions such as path patching, which involve selecting particular inputs, choosing a subset of components that will be patched, and then observing changes in model behavior. Here, we show that several important aspects of the detokenization stage can be understood purely by analyzing model weights, without performing any model inference steps. Specifically, we introduce an analytical decomposition of first-layer attention in GPT-2. Our decomposition yields interpretable terms that quantify the relative contributions of position-related, token-related, and mixed effects. By focusing on terms in this decomposition, we discover weight-based explanations of attention bias toward close tokens and attention for detokenization.

github.com/gokamoda/lm-detokenization

1 Introduction

Language models (LMs) (Vaswani et al., 2017; Radford et al., 2019; Brown et al., 2020; Dubey et al., 2024; Gemma Team, 2024) operate on sequences of subword tokens (Kudo, 2018; Sennrich et al., 2016). Consequently, LMs encounter many words and names, e.g., "Libertarian", not in their natural form but split into parts such as "Liber" and "tarian". Since such subword sequences are not necessarily linguistically meaningful, it is believed that a core function of early LM layers is to *detokenize* (Elhage et al., 2022) sequences of subword tokens into more meaningful representations of words and names that, taken together, form the LM's *inner vocabulary* (Kaplan et al., 2024). This inner vocabulary contains the basic meaning representations on which subsequent stages of inference operate (Lad et al., 2024). However, evidence for the detokenization hypothesis has so far only been collected from empirical experiments that require selecting specific inputs and/or training probes in order to localize layers showing behavior consistent with detokenization (Gurnee et al., 2023; Kaplan et al., 2024). Here, we take an alternative approach. By developing a new decomposition of first-layer attention in GPT-2, we show that several important aspects of detokenization can be understood from model weights alone, without training probes or performing any inference steps (Fig. 1).

A crucial part of detokenization is attention to tokens that, taken together, comprise a word or phrase. Prior work has analyzed this n-gram attention aspect of detokenization (Gurnee et al., 2023; Kaplan et al., 2024). However, these analyses did not disentangle token content effects from positional biases. Although not in the context of detokenization, Dar et al. (2023) and Elhage et al. (2021) analyze the interaction between pairs of vocabularies in the attention layer of GPT-2 and newly-trained model, respectively. However, they do not take into account the effect of LayerNorm. Going beyond prior work, we analyze the effect of token representation without positional information on attention weights while fully considering LayerNorm.

While such token-deriving attention is an important aspect, it is not sufficient for performing detokenization. Another important aspect is attention to close tokens. We conduct weight-based analysis and show that higher attention is assigned to positionally close tokens in the first layer of GPT-2 (Radford et al., 2019), regardless of the input token (Sections 5.1 to 5.3). In particular, we identify two components in the learned absolute position embeddings The first component can be

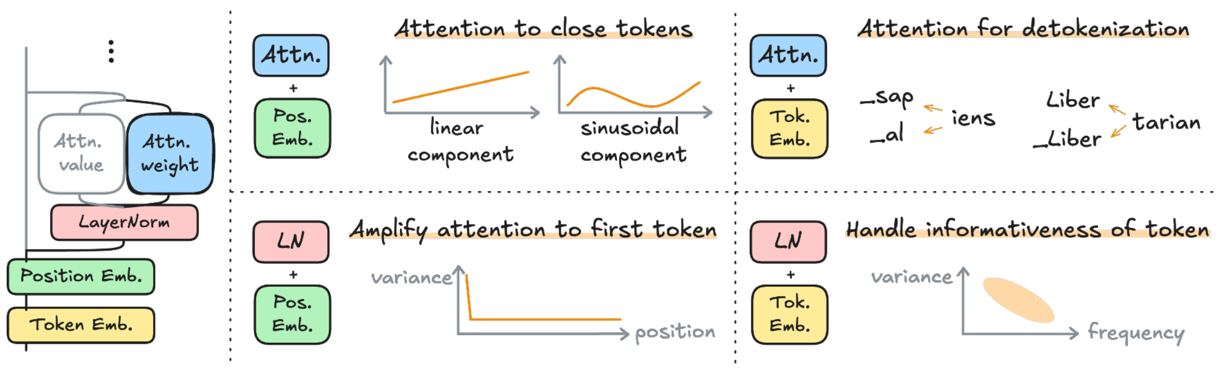


Figure 1: Focusing on the token/position embeddings, first LayerNorm layer, and the first attention layer, we conduct weight analyses and show high attention scores are assigned to close (top middle; Sections 5.1 to 5.3) *and* related (top right; Section 4), supporting the detokenization hypothesis. We also show that the high attention score to the first token derives from LayerNorm (bottom middle; Section 6.4). Regarding token embedding, we also discuss the relationship between token frequency and LayerNorm (bottom right; Section 6.3).

seen as a linear bias component, similar to ALiBi (Press et al., 2022). The second component has a sinusoidal shape, which is reminiscent of sinusoidal (Vaswani et al., 2017) and rotary (Su et al., 2024) position encoding. In superposition, these two components induce an attention bias towards close tokens.

In summary, we analyze the first attention layer in GPT-2, separating computations deriving from position embedding and token embedding. As a whole, our results comprehensively explain and support the mechanism of detokenization in that attention layers attend to related tokens that are positioned close.

More broadly, we worked on the internal understanding of the model inference without selecting prompts and running inference. This paper shows the first findings in this view, providing more theoretical proof of underlying mechanisms.

2 Background

Early layers of LMs are hypothesized to *detokenize* over-segmented words and phrases. In this section, we briefly provide the necessary background on the detokenization hypothesis. We also discuss relevant positional encoding schemes since positional information plays an important role during detokenization.

2.1 Detokenization

Elhage et al. (2022) introduce *de-tokenization* to explain that initial layers of the language model they trained contribute to mapping multi-token or compound words to a "semantic representation". For example, they found neurons responding to "Libertarian", which was tokenized into "Libert" and "arian". Lad et al. (2024) use this term as the name of the hypothesis ("detokenization hypothesis") for the first stage of inference where language models "integrate local context to convert raw token representations into coherent entities". Based on the detokenization hypothesis, Kaplan et al. (2024) collect multi-token words, input them to a model, and use patch scope technique (Ghandeharioun et al., 2024) to inspect if original tokens can be recovered from a single hidden state between intermediate layers. Gurnee et al. (2023), focusing on MLP neuron activations, train probes that distinguish n-grams, finding "compound word neurons". Geva et al. (2023), although they do not mention "detokenization", report that language models build multi-token subject representations, such as "Beats Music", from raw token representations in the early layers.

In this work, we show some aspects of the detokenization process that can be understood just from the weights of the target LM alone, without running a single forward or backward pass.

2.2 Positional Encoding

Transformer-based language models explicitly use architectures that catch positional information. Vaswani et al. (2017) originally employed "positional encoding" defined by sine and cosine functions of different frequencies and added a vector representing the absolute token position at the embedding layer. Models like BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) use similar architecture, except the position embeddings are learned through pre-training. Yamamoto and Matsuzaki (2023) report that RoBERTa, an encoder model, learns sinusoidal position embedding through training, which contributes to assigning high attention scores to close tokens. GPT-2 (Radford et al., 2019) also uses learned absolute position embedding, but they leave analyses of causal models, which showed different trends from encoder-decoder models in their analyses, to future work.

As another method for positional encoding, ALiBi "biases query-key attention scores with a penalty that is proportional to their distance" (Press et al., 2022). We show in Section 5.1 that GPT-2 learns both the sinusoidal position bias, which relates with Yamamoto and Matsuzaki (2023) and the linear bias which relates with ALiBi.

3 Decomposing the First Attention Layer

In this study, we select GPT-2 as our model of analysis, following prior work (Lad et al., 2024; Dar et al., 2023), which investigates detokenization and/or attention trends. GPT-2 is also a model that is often analyzed in prior works focusing on model interpretability in general (Geva et al., 2023; Hanna et al., 2023; Conmy et al., 2023).

Before conducting weight analyses, we redefine some of the weights to make analyses simple. Specifically, focusing on LayerNorm and attention, we fold in multiple linear transformations into a single linear transformation and ignore terms that can be ignored. We describe our analysis settings in the following subsections, showing it is mathematically equivalent to the original computation in the GPT-2 model.[1]

3.1 Embedding

In GPT-2's Embedding layer, the initial hidden state is computed based on the Token ID and the absolute position of the token. Let the Token ID of the i-th token be denoted as ID_i, and the Token Embedding Matrix as $\boldsymbol{E} \in \mathbb{R}^{|V| \times d}$, where $|V|$ is the size of the vocabulary, and d is the embedding dimension. Additionally, GPT-2 employs absolute position embeddings, with the position embedding matrix denoted as $\boldsymbol{P} \in \mathbb{R}^{L \times d}$, where L is the maximum sequence length. The output of the Embedding layer at position i, $\boldsymbol{x}_i$, is thus represented as follows:

$$\boldsymbol{x}_i = \boldsymbol{e}_{\mathrm{ID}_i} + \boldsymbol{p}_i \tag{1}$$

3.2 Layer Normalization

The Transformer architecture applies layer normalization at various points. The specific form of layer normalization used in GPT-2, namely LayerNorm can be expressed [2] as follows:

$$\underline{\mathrm{LN}}(\boldsymbol{x}) := \frac{\boldsymbol{x}}{\sigma(\boldsymbol{x})}\left(\boldsymbol{I} - \frac{1}{d}\mathbf{1}^\top\mathbf{1}\right)\mathrm{diag}(\boldsymbol{\gamma}) + \boldsymbol{\beta} \tag{2}$$

$$\sigma(\boldsymbol{x}) := \sqrt{\mathrm{Var}(\boldsymbol{x}) + \epsilon} \tag{3}$$

where ϵ is a small constant added to prevent division by zero, and $\boldsymbol{\gamma}, \boldsymbol{\beta} \in \mathbb{R}^d$ are learnable parameters. Thus, once division by $\sigma(\boldsymbol{x})$ has been computed, LayerNorm is simply an affine transformation. Originally introduced to stabilize and speed up model training (Ba et al., 2016), from the perspective of interpretability, this component is mainly viewed as a nuisance factor that complicates analysis due to the appearance of the variance term $\mathrm{Var}(\boldsymbol{x})$ taken over the hidden state $\boldsymbol{x}$. While interpretability work has dealt with LayerNorm by "folding" the affine transformation part into other components (Nanda, 2023), ignoring it (Dar et al., 2023), or removing it altogether (Heimersheim, 2024), we will show that the LayerNorm variance plays an important role by in effect acting as an if-then condition on token positions.

3.3 Attention

The role of attention is to dynamically mix contextual information into the representation of the current token. In a causal model, given current position i and the context information $\boldsymbol{X}$, the attention layer with H heads performs the following computation:

$$\mathrm{ATTN}(i, \boldsymbol{X}) := \sum_{h=1}^{H}\sum_{j=1}^{i} \alpha_{i,j,h}\boldsymbol{v}_h(\boldsymbol{x}_j)\boldsymbol{W}_h^O + \boldsymbol{b}^O \tag{4}$$

where $\boldsymbol{v}_h(\boldsymbol{x}_j)$ is the value vector of dimension $d' = d/H$ associated with token representation $\boldsymbol{x}_j$, matrix $\boldsymbol{W}^O$ and vector $\boldsymbol{b}^O$ are the weight and bias of the affine output transformation, and the attention weights $\alpha_{i,j,h}$ from token position i to j in head h with $\sum_j \alpha_{i,j,h} = 1$ are given by:

$$\alpha_{i,j,h} := \operatorname*{softmax}_{\boldsymbol{x}_j \in \boldsymbol{X}, j \le i}\left(s_{i,j,h}/\sqrt{d'}\right) \tag{5}$$

[1]We provide the detailed notation and decomposition in Appendices A and B

[2]We underline symbols from the original formulation to distinguish them from symbols in our reformulation in 3.4

Here, $s_{i,j,h}$ are unnormalized attention scores:

$$\underline{s_{i,j,h}} := \boldsymbol{q}_h(\boldsymbol{x}_i)\boldsymbol{k}_h(\boldsymbol{x}_j)^\top \tag{6}$$

where $\boldsymbol{q}_h$ and $\boldsymbol{k}_h$ are affine transformation from an input $\boldsymbol{x}$ to a query vector and key vector:

$$\boldsymbol{q}_h(\boldsymbol{x}) := \boldsymbol{x}\boldsymbol{W}_h^Q + \boldsymbol{b}_h^Q \tag{7}$$
$$\boldsymbol{k}_h(\boldsymbol{x}) := \boldsymbol{x}\boldsymbol{W}_h^K + \boldsymbol{b}_h^K \tag{8}$$

3.4 Reformulating LayerNorm and Attention

From Eqs. 4 and 5, it can be observed that in the computation of ATTN, input $\boldsymbol{x}$ are always first subjected to an affine transformation. Because the input to ATTN is the output of LN, the linear part of $\underline{\text{LN}}$ can be absorbed into the affine transformations for the $\boldsymbol{q}_h$, $\boldsymbol{k}_h$ and $\boldsymbol{v}_h$ functions in ATTN. Therefore, instead of Eq. 2, we redefine LN:

$$\text{LN}(\boldsymbol{x}) := \boldsymbol{x}/\sigma(\boldsymbol{x}) \tag{9}$$

We also redefine the affine transformation that is applied to $\boldsymbol{q}_h$ in Eq. 7, with analogous redifintions for $\boldsymbol{k}_h$ and $\boldsymbol{v}_h$:

$$\boldsymbol{W}_h^Q := \left(\boldsymbol{I} - \frac{1}{d}\boldsymbol{1}^\top\boldsymbol{1}\right)\text{diag}(\gamma)\underline{\boldsymbol{W}_h^Q} \tag{10}$$
$$\boldsymbol{b}_h^Q := \beta\underline{\boldsymbol{W}_h^Q} + \underline{\boldsymbol{b}}_h^Q \tag{11}$$

Next, we reformulate the unnormalized attention scores $\underline{s_{i,j,h}}$ defined in Eq. 6:

$$\underline{s_{i,j,h}} = \boldsymbol{q}_h(\boldsymbol{x}_i)\boldsymbol{W}_h^{K\top}\boldsymbol{x}_j^\top + \boldsymbol{q}_h(\boldsymbol{x}_i)\boldsymbol{b}_h^{K\top} \tag{12}$$

Note that in Eq. 5, the softmax is applied over token positions j, while the second term in Eq. 12 does not depend on j, i.e., when computing the attention from token position i to other tokens, this term is a constant. Since the softmax is invariant to the addition of a constant, this term can be ignored in analysis, and by extension, this means that the bias $\boldsymbol{b}_h^K$ is actually completely meaningless. By further expanding Eq. 12, we obtain:

$$s_{i,j,h} := \boldsymbol{x}_i\boldsymbol{W}_h^{QK}\boldsymbol{x}_j^\top + \boldsymbol{b}_h^{QK}\boldsymbol{x}_j^\top \tag{13}$$

with

$$\boldsymbol{W}_h^{QK} := \boldsymbol{W}_h^Q\boldsymbol{W}_h^{K\top} \tag{14}$$
$$\boldsymbol{b}_h^{QK} := \boldsymbol{b}_h^Q\boldsymbol{W}_h^{K\top} \tag{15}$$

This reformulation of the attention score computation is the basis for our decomposition, which we will perform following subsection.

3.5 Decomposition

The first term in Eq. 13 depends on the hidden state of the present token $\boldsymbol{x}_i$ and the hidden state of the past token $\boldsymbol{x}_j$. That is, this term becomes large when the two hidden states $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are similar under the linear projection $\boldsymbol{W}_h^{QK}$, which is why we will call this term the "comparison term". The second term in Eq. 13 depends only on the hidden state of the past token $\boldsymbol{x}_j$ and is independent of the present token i. A large $\boldsymbol{b}_h^{QK}\boldsymbol{x}_j^\top$ value means, figuratively, that token j self-asserts its relevance regardless of context, which is why call this term the "self-assertion term".

So far we have ignored positional encoding. The input to the first attention layer consists of the token embedding $\boldsymbol{e}_{\text{ID}_i}$ and position embedding $\boldsymbol{p}_i$:

$$\boldsymbol{x}_i := \frac{\boldsymbol{e}_{\text{ID}_i} + \boldsymbol{p}_i}{\sigma(\boldsymbol{e}_{\text{ID}_i} + \boldsymbol{p}_i)} \tag{16}$$

By plugging Eq. 16 into Eq. 13 and expanding, we obtain a decomposition of the first layer's attention scores into six terms:

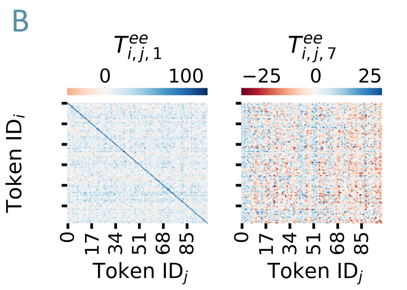
$$\begin{aligned} s_{i,j,h} = {} & \underset{T^{\text{ee}}_{i,j,h}}{\frac{\boldsymbol{e}_{\text{ID}_i}\boldsymbol{W}_h^{QK}\boldsymbol{e}_{\text{ID}_j}^\top}{\sigma_i\sigma_j}} + \underset{T^{\text{pp}}_{i,j,h}}{\frac{\boldsymbol{p}_i\boldsymbol{W}_h^{QK}\boldsymbol{p}_j^\top}{\sigma_i\sigma_j}} \\ & + \underset{T^{\text{pe}}_{i,j,h}}{\frac{\boldsymbol{p}_i\boldsymbol{W}_h^{QK}\boldsymbol{e}_{\text{ID}_j}^\top}{\sigma_i\sigma_j}} + \underset{T^{\text{ep}}_{i,j,h}}{\frac{\boldsymbol{e}_{\text{ID}_i}\boldsymbol{W}_h^{QK}\boldsymbol{p}_j^\top}{\sigma_i\sigma_j}} \\ & + \underset{T^{\text{e}}_{j,h}}{\frac{\boldsymbol{b}_h^{QK}\boldsymbol{e}_{\text{ID}_j}^\top}{\sigma_j}} + \underset{T^{\text{p}}_{j,h}}{\frac{\boldsymbol{b}_h^{QK}\boldsymbol{p}_j^\top}{\sigma_j}} \end{aligned} \tag{17}$$

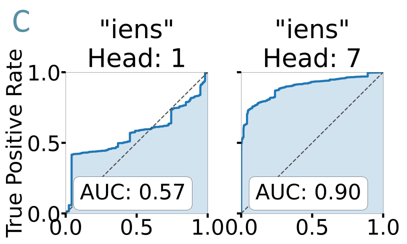
where $\sigma_i := \sigma(\boldsymbol{e}_{\text{ID}_i} + \boldsymbol{p}_i)$. For brevity, we will refer to each of the six terms in Eq. 17 using the underset blue alias.

With this decomposition in place, we are now ready to analyze specific terms. Starting with the token comparison term T^{ee}, we will show that it can be understood as representing token-token affinities and use it to identfy detokenization heads (Section 4). Then we show how the two purely position-related terms T^{p} and T^{pp} contribute to detokenization by biasing attention towards preceding tokens (Section 5). Finally, we analyze the remaining three terms, which are not directly related to detokenization, in Section 6.

A

Head	Token i	Token j (rank)	Detokenization
4	iens	_sap (1)	_sapiens
4	iens	_Sap (3)	_Sapiens
7	iens	_sap (1)	_sapiens
7	iens	Al (2)	Aliens
7	iens	_al (5)	_aliens
4	tarian	Liber (1)	Libertarian
4	tarian	_Liber (2)	_Libertarin
7	_Jackson	_Peter (1)	_Peter_Jackson
7	_Jackson	_Jesse (2)	_Jesse_Jackson
7	_Jackson	_Michael (3)	_Michael_Jackson
7	_Jackson	Michael (4)	Michael_Jackson
7	_Jackson	Peter (5)	Peter_Jackson





D

Head	AUROC
7	0.88
11	0.81
6	0.79
0	0.73
4	0.69
3	0.63
10	0.62
2	0.55
8	0.44
1	0.40
9	0.40
5	0.29

Figure 2: **A**: Examples of support for detokenization. When the current position token is “iens”, the past token that yields the largest T^{ee} value (=Rank 1) is “_sap” in head#4 and head#7. **B**: Heatmap of T^{ee} for head#7 and head#1. Tokens are randomly sampled from the vocabulary for visualization. **C**: ROC of head#7 and head#1 when token i is “iens”. **D**: Average AUROC for each head. Heads with high AUROC values contribute to the reconstruction of bi-grams, consequently contributing to detokenization.

4 Detokenization and Token Affinity

The token comparison term T^{ee} is highly relevant to detokenization because its numerator $\boldsymbol{e}_{\text{ID}_i}\boldsymbol{W}_h^{QK}\boldsymbol{e}_{\text{ID}_j}^\top$ compares the embedding of the current token $\boldsymbol{e}_{\text{ID}_i}$ and the past token $\boldsymbol{e}_{\text{ID}_j}$ through the linear transformation $\boldsymbol{W}_h^{QK}$. A large T^{ee} value means that the source token is biased to pay high attention to the target token.

In this section, we show examples of detokenization performed by T^{ee} (Section 4.1) and investigate which heads actually contribute to detokenization (Section 4.2).

4.1 Examples of Detokenization

Here, we show some examples of detokenization, where attention heads assign high T^{ee} value to tokens that form words or two-token entities together with source (i.e. current) token.

First, we compiled some words or entities that are split into two tokens by the GPT-2 tokenizer (e.g., “sapiens”). For each instance, we fix the second token (“iens”) id as ID$_i$ and compute T^{ee} against all ID$_j$ in vocabulary V in all heads. In Fig. 2-A, we show pairs of ID$_j$ and ID$_i$ with high T^{ee} that together form meaningful sequences. For example, when ID$_i$ is “iens”, token “_sap” and “Al” yields the two largest T^{ee} scores among 50,257 subwords in head#7, detokenizing “_sapiens” and “Aliens”. In Appendix Table 1, we show other examples of T^{ee} contributing to detokenizing words, people’s names, or chemical substances.

4.2 Which Heads Perform Detokenization?

Before conducting a detailed analysis, we visualize the T^{ee} matrix for each of the 12 attention heads as a heatmap to gain an overview (Fig. 2-B, Fig. 12). This visualization reveals that the heads can be broadly divided into two categories: those with diagonal lines in the heatmap and those without. Head#1, with scores in the range (-47, 127), has a clear diagonal line, indicating that the attention score deriving from T^{ee} is the highest when the target token is identical to the current past token. Head#7 on the other hand, does not show such a diagonal line. Instead, scores in range (-35, 31) are distributed across the heatmap.

What do these two trends suggest? Detokenization typically requires attending to different tokens – for example, paying attention from “iens” to “sap” to reconstruct the word “sapiens,” rather than attending from “iens” to “iens.” Therefore, heads that exhibit clear diagonal lines, such as head#1, are less likely to contribute to detokenization compared to heads with more dispersed attention patterns, such as head#7.

Next, we investigate the degree to which each attention head contributes to detokenization by inspecting the relationship between the scores of T^{ee} and bi-gram frequency.

For a fixed token at position i, we first compute

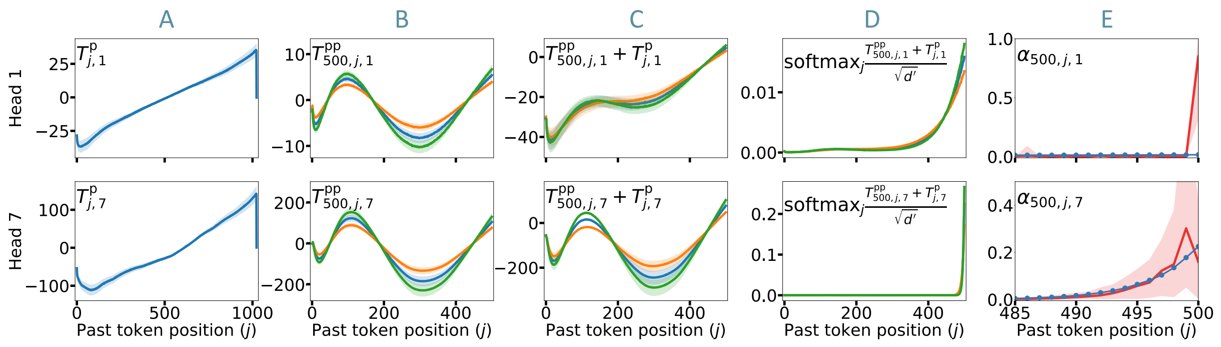


Figure 3: A: T^{p}_j for all context token position j for head#1 (top) and #7 (bottom). The shaded area represents the variance of the term deriving from e_{ID_i} Y-axis titles in this, and subsequent figures are inset for readability. B: $T^{\mathrm{pp}}_{500,j} = \boldsymbol{p}_i \boldsymbol{W}^{QK}_h \boldsymbol{p}^{\top}_j / \sigma_i \sigma_j$ for context token position $j \leq 500$. The blue, green, and orange lines show scores with mean, maximum, and minimum standard deviation for σ_j: $\sigma_j = \frac{1}{|V|} \sum_{\mathrm{ID}} \sigma(\boldsymbol{e}_{\mathrm{ID}} + \boldsymbol{p}_j)$, $\sigma_j = \max_{\mathrm{ID}} \sigma(\boldsymbol{e}_{\mathrm{ID}} + \boldsymbol{p}_j)$, and $\sigma_j = \min_{\mathrm{ID}} \sigma(\boldsymbol{e}_{\mathrm{ID}} + \boldsymbol{p}_j)$, respectively. C,D: Sum of $T^{\mathrm{pp}}_{500,j}$ and T^{p}_j and its result after taking softmax with a temperature of $\sqrt{d'}$. E: Empirical attention weights aggregated over texts in OpenWebText Corpus when present token position $i = 500$ focusing on the last few j positions. The red line and area show the empirically observed weights and the blue line corresponds with the blue lines in D.

T^{ee} against all 50,257 tokens in the vocabulary for GPT-2. Using bi-gram counts of OpenWebText Corpus (Gokaslan et al., 2019), we compute AUROC with True Positive Rate defined as the proportion of bi-gram counts with T^{ee} above a threshold. Fig. 2-C shows the Recall curve for head#7 and head#1 when the token at position i is fixed to “iens” or “Jackson”. High AUC (Area Under Curve) indicates that T^{ee} with high scores are likely to reconstruct frequent bi-grams. By computing AUC for all position i tokens and taking the average, we quantify how each attention head contributes to reconstructing bi-grams. Fig. 2-D shows the result, showing largest AUROC in head#7 which was also observed in Fig. 2-A and Table 1, supporting contribution to detokenization.

5 Detokenization and Token Positions

Another important aspect is attention to close tokens. We conduct weight-based analysis on T^{p} (Section 5.1) and T^{pp} (Section 5.2) and identify two different trends: one linear and one sinusoidal. Then we show in Section 5.3 show that the sum of the two components biases attention to positionally close tokens. Finally, we verify the positional bias empirically in Section 5.4.

5.1 Linear Self-assertion Term (T^{p})

We visualize the position-deriving self-assertion term T^{p} for head#1 and head#7 in Fig. 3-A [3]. Since GPT-2 is a causal model, the causal mask ensures that no context after the present token is referenced. For example, when $i = 200$, the part of Fig. 3-A to the right of $j = 200$ is ignored. In the range $0 \leq j \leq 200$, since the self-assertion term is monotonically increasing with respect to j, we can observe that high attention scores are assigned to tokens that are relatively positionally close. The same applies for almost all i, thereby constituting a bias towards high attention scores to nearby tokens. [4]

5.2 Sinusoidal Comparison Term (T^{pp})

We visualize the position-deriving comparison term T^{pp} for head#1 and head#7 in Fig. 3-B when $i = 500$[5]. A notable difference from Fig. 3-A is that this term exhibits undulating patterns, which could be related to the observations by Yamamoto and Matsuzaki (2023). However, what is similar is that this term also assigns high attention scores to close tokens, contributing to high attention score on nearby tokens.

5.3 Sum of T^{pp} and T^{p}

Fig. 3-C shows the sum of the two terms deriving from position embedding[6]. The combination of the monotonic increase in the T^{p} and the sinusoidal component from the T^{pp} leads to high attention scores being assigned to tokens positioned nearby.

The result after applying softmax function to the sum of the two terms, is shown in Fig. 3-D. It in-

[3]Results for other heads are in appendix Fig. 8.

[4]We show analyses of the exceptional behavior shown in the first and last few positions in Section 6.4.

[5]Results for other heads and i are in appendix Fig. 9.

[6]Results for other heads and i are in Appendix Fig. 10

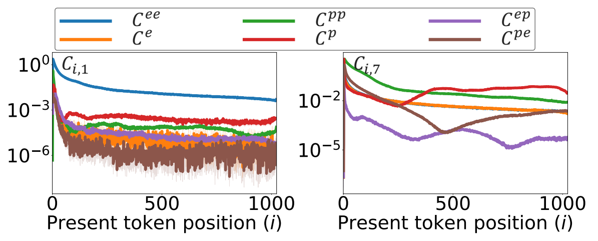


Figure 4: Contribution of the 6 terms in Eq. 17 for each current token position i for head#1 and #7

dicates that the attention weight is concentrated on nearby tokens, and the sinusoidal component originating from the T^{pp} is almost entirely suppressed.

5.4 Empirical Verification

To verify whether the observations through the weight analyses hold true empirically, we use natural language texts from OpenWebText Corpus (Gokaslan et al., 2019) and obtain $\alpha_{i,j,h}$ from the first attention layer. Fig. 3-E shows the results when $i = 500$. Head#7 shows a pattern similar to the plot in Fig. 3-D (also shown in blue line), with high attention to the nearest tokens. In contrast, head#1 predominantly attends to itself (Blue line, $j = 500$), which differs from the red line (also shown in Fig. 10). The discrepancy can be attributed to the dominance of T^{ee}. We showed in Fig. 2-B that head#1 assigns high attention scores to the identical token, with scores in range (-50, 100). Furthermore, from Fig. 3-B, it can be observed that head#1 assigns scores within a narrower range (-40, 0) across broad positions.

6 Followup Experiments

6.1 Contribution of the Six Terms to Attention Weights

In Sections 4 and 5, we conducted weight analyses of T^{ee}, T^{p}, and T^{pp}. Here, we inspect the contribution of all 6 terms in Eq. 17 to check if we are missing any crucial terms. We use KL-Divergence as a metric. For example, we define the contribution of the T^{ee}, c^{ee} as follows:

$$c^{\text{ee}}_{i,h} = D_{\text{KL}}(P_{i,h}||Q_{i,h}) \tag{18}$$

$$Q_{i,h} = \begin{bmatrix}\alpha_{i,0,h} & \cdots & \alpha_{i,i,h}\end{bmatrix} \tag{19}$$

$$P_{i,h} = \begin{bmatrix}\alpha'_{i,0,h} & \cdots & \alpha'_{i,i,h}\end{bmatrix} \tag{20}$$

$$\alpha'_{i,j,h} = \underset{\boldsymbol{x}_j \in \boldsymbol{X}, j \leq i}{\text{softmax}} \left(\frac{s_{i,j,h} - T^{\text{ee}}_{i,j,h}}{\sqrt{d'}}\right) \tag{21}$$

Fig. 4 shows that the three terms analyzed in Sections 4 and 5 have relatively high contributions

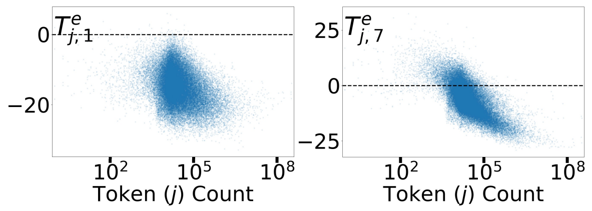


Figure 5: Relation between token frequency and T^{e} for head#1 and head#7.

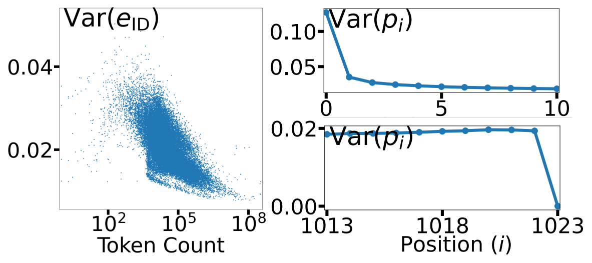


Figure 6: **Left**: Relationship between the variance of each token embedding and their corpus counts. **Right**: Variance of all first 10 (top), and last 10 (bottom) position embeddings.

to the attention weight computation. From the three remaining terms, T^{pe} and T^{ep}, contribute less. However, T^{e} shows a high contribution. We analyze this term in the following section.

6.2 Token Self-assertion Term (T^{e})

T^{e} could be interpreted as a bias term for which token to attend to. Here, we use token frequency as a proxy for the informativeness of a token and plot the relationship between token count and T^{e} in Fig. 5[7]. We used the OpenWebText Corpus (Gokaslan et al., 2019) to get token counts. The results show that several heads exhibit a high correlation with token frequency. For example, the Spearman correlation coefficient between T^{e} and token frequency for head#7 is -0.68, while for head#1, it is -0.31.

6.3 Variance of token embeddings

The variance of token embeddings plays a critical role in LayerNorm (Eq. 9). Fig. 6-left shows the relationship between token count and variance of the corresponding embedding vector. The Spearman correlation coefficient of this relation is -0.63, indicating the variance of rare tokens is high. The variance of the norm of each token embedding vector is 0.19. After dividing by the root of the variance, the variance becomes 0.00. That is, when

[7]We show the results for other heads in Fig. 11.

applying LayerNorm directly to the token embeddings it has the effect of equalizing the norms of all token embeddings[8]. Oyama et al. (2023) shows that the norm of word embedding encodes information gain. However, as shown in Fig. 5, even after passing through LayerNorm, information on token count can be extracted. Though it may depend on the definition of information gain, we speculate that something akin to information gain is encoded in places other than the norm.

6.4 Variance of Position Embeddings

We visualize the variance of the position embeddings, which plays a critical role in LayerNorm when visualizing over context position j as suggested by Eqs. 3 and 9 and Section 3.5, in Fig. 6-right. For most positions, the variance remains constant. However, there are two exceptions: the variance for the first token is exceptionally high, and the variance for the last token is significantly low. As a result, the variance of the position embeddings takes the shape of a step function.

The large variance at position 0, combined with the negative attention score without LayerNorm at $j = 0$, leads to an exceptionally high attention score. It creates a non-linear function implementing an "if-then" condition: if the attention is directed towards the first token of the context, the attention weight is amplified. We discuss why such "switch" is implemented and why the variance of the last position takes exceptional value in Section 6.4.

Exceptionally High Variance at First Position Gurnee et al. (2024) reports that the norm of the value vector for the BOS token of GPT-2-medium is 19.4 times smaller than the average for other tokens and looks for neurons that may utilize the BOS token as *attention sink* (Xiao et al., 2023) via ablation studies. We propose an alternative interpretation of these findings: by exceptionally increasing the variance of the first position embedding, the first token is used as an attention sink. However, the attention sink may not be used depending on the context length and the head. In fact, while Fig. 3-**A** has a high attention score assigned to the first token, Fig. 3-**D** has the score suppressed[9]. It can be interpreted that the mechanism is embedded in the LayerNorm term common to all heads and all current positions i because, as the context length increases, the scores for nearby tokens, mainly due to T^{p}, exceed the scores for the first token, rendering the attention sink possibly unnecessary and ignorable by the softmax function.

[8]Constant variance of position embedding (Section 6.4) and small mean absolute covariance of position embedding and token embedding (46 times smaller than the mean variance of token embedding) supports the validity of this observation.

[9]Refer to Fig. 10 for other variations.

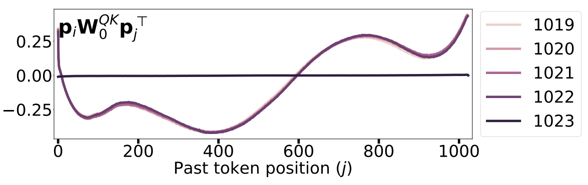


Figure 7: $T^{\text{pp}}_{i,j,0}$ without LayerNorm for $1019 \leq i \leq 1023$. When the current position i is 1023, the maximum input length of GPT-2, the attention scores show a distinct outlier behavior.

Exceptionally Low Variance at Last Position A plausible explanation for the irregularity observed at the last position in Fig. 6-bottom-right is undertraining. Fig. 7 visualizes the T^{pp} without LayerNorm for the final positions, specifically for $1019 \leq i \leq 1023$. While the plots largely overlap for $1019 \leq i \leq 1022$, a distinct deviation can be seen at $i = 1023$, where the score remains consistently flat around zero (black line). We suspect that this phenomenon occurs because the maximum length for prompts accepted by GPT-2 is 1024 tokens, and the model was not trained with the loss for the 1025th token.

7 Conclusions

We analyzed the first attention layer of GPT-2 to investigate the detokenization mechanism. Considering detokenization to be a phenomenon that occurs when deeply related tokens are close together, we conducted analyses that separate token embedding and position embedding to provide multiple theoretical supports for detokenization. First, we showed that the self-assertion attention term and comparison term derived from position embedding contribute to assigning high attention to relatively close tokens. Furthermore, we suggested that position embedding and LayerNorm are deeply related to the phenomenon of high attention being assigned to the first token. Regarding the relevance between tokens, we showed that the comparison term deriving from token embedding contributes to this, and obtained results that are generally consistent with existing research.

Limitations

In this study, we analyzed the weights learned by the first layer of GPT-2.

The detokenization mechanism itself does not occur only in the first layer, thus further analysis of other early layers may be necessary for a deeper analysis. However, by analyzing a small version of GPT-2, a 12-layer model, we may have captured the main mechanism of detokenization even with only one layer.

The analysis we conducted, which separates the attention weight calculation into token embedding and position embedding, was possible and meaningful because GPT-2 adopts Learned Absolute Position Embedding. For other models that use ALiBi or Rotary Position Embedding, it may be apparent that models attend to close tokens as the parameters in these methods are defined prior to training. We believe our contribution lies in that we show that even without an explicit design of position bias, models can learn to attend to close tokens and contribute to detokenization from the first layer.

Furthermore, the self-assertion term is a term that appears because the transformation for calculating the key and query vectors is an affine transformation with a bias. However, recent models such as Llama 3 (Dubey et al., 2024) and Gemma 2 (Gemma Team, 2024) do not use bias terms in linear layers. While this means analyses breaking down attention into self-assertion terms and comparison terms cannot be done on such models, we believe that our results also raise the question of whether bias terms can really be eliminated.

Regarding the choice of subject model for analysis, our ultimate goal is to gain insights into robust and efficient models. Therefore, we believe that while the subject model must achieve a certain level of performance, it is not necessary to analyze only the SOTA models. In addition, we believe that designing and analyzing models that are easier to interpret is one direction to take in the context of the recent discussion on the reliability of language models.

Another limitation is that our analysis explains only a part of the detokenization process. Detokenization is not just attending to close preceding tokens, but *selectively* doing so, since single-token words do not require detokenization. So the "attend to close preceding tokens" mechanism should be "switched off" for these tokens, and it is one of the future works to offer a weight-based explanation of how and where this "switch" is implemented.

Ethics Statement

Our analyses involved a pretrained language model, GPT-2 (Radford et al., 2019) (124M parameter model from Hugging Face[10], MIT License), and corpus, OpenWebText, which may be characterized by various forms of social biases. We analyze how GPT-2 processes natural language inputs, which is within the scope of its intended usage.

Acknowledgements

This work was supported by JST/CREST (JPMJCR20D2) and JST/BOOST (JPMJBS2421). We would also like to thank the members of TohokuNLP for their frequent participation in discussions during the course of this research.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer Normalization. *arXiv [stat.ML]*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, and others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901.

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.

Guy Dar, Mor Geva, Ankit Gupta, and Jonathan Berant. 2023. Analyzing transformers in embedding space. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16124–16170. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North*, pages 4171–4186. Association for Computational Linguistics.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, and others. 2024. The Llama 3 herd of models. *arXiv [cs.AI]*.

Nelson Elhage, Tristan Hume, Catherine Olsson, Neel Nanda, Tom Henighan, Scott Johnston, Sheer

[10]https://huggingface.co/openai-community/gpt2

El Showk, and others. 2022. Softmax Linear Units. https://transformer-circuits.pub/2022/solu/index.html.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Mann Ben, Amanda Askell, and others. 2021. A Mathematical Framework for Transformer Circuits. https://transformer-circuits.pub/2021/framework/index.html.

Gemma Team. 2024. Gemma 2: Improving open language models at a practical size. *arXiv [cs.CL]*.

Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. 2023. Dissecting recall of factual associations in auto-regressive language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12216–12235. Association for Computational Linguistics.

Asma Ghandeharioun, Avi Caciularu, Adam Pearce, Lucas Dixon, and Mor Geva. 2024. Patchscopes: A unifying framework for inspecting hidden representations of language models. In *Forty-first International Conference on Machine Learning*.

Aaron Gokaslan, Vanya Cohen, Ellie Pavlick, and Stefanie Tellex. 2019. OpenWebText Corpus. http://Skylion007.github.io/OpenWebTextCorpus.

Wes Gurnee, Theo Horsley, Zifan Carl Guo, Tara Rezaei Kheirkhah, Qinyi Sun, Will Hathaway, Neel Nanda, and Dimitris Bertsimas. 2024. Universal neurons in GPT2 language models. *Transactions on Machine Learning Research*, 2024.

Wes Gurnee, Neel Nanda, Matthew Pauly, Katherine Harvey, Dmitrii Troitskii, and Dimitris Bertsimas. 2023. Finding neurons in a haystack: Case studies with sparse probing. *Transactions on Machine Learning Research*.

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*.

Stefan Heimersheim. 2024. You can remove GPT2's LayerNorm by fine-tuning. *arXiv [cs.CL]*.

Guy Kaplan, Matanel Oren, Yuval Reif, and Roy Schwartz. 2024. From tokens to words: On the inner lexicon of LLMs. In *The Thirteenth International Conference on Learning Representations*.

Taku Kudo. 2018. Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75.

Vedang Lad, Wes Gurnee, and Max Tegmark. 2024. The Remarkable Robustness of LLMs: Stages of Inference? In *ICML 2024 Workshop on Mechanistic Interpretability*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv [cs.CL]*.

Neel Nanda. 2023. TransformerLens: A library for mechanistic interpretability of GPT-style language models.

Momose Oyama, Sho Yokoi, and Hidetoshi Shimodaira. 2023. Norm of word embedding encodes information gain. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2108–2130. Association for Computational Linguistics.

Ofir Press, Noah Smith, and Mike Lewis. 2022. Train Short, Test Long: Attention with Linear Biases Enables Input Length Extrapolation. In *International Conference on Learning Representations*.

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

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. RoFormer: Enhanced transformer with Rotary Position Embedding. *Neurocomputing*, 568(127063):127063.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. *Advances in Neural Information Processing Systems*, 30.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2023. Efficient Streaming Language Models with Attention Sinks. In *The Twelfth International Conference on Learning Representations*.

Yuji Yamamoto and Takuya Matsuzaki. 2023. Absolute position embedding learns sinusoid-like waves for attention based on relative position. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 15–28. Association for Computational Linguistics.

A Notation

$$\boldsymbol{E} := \begin{bmatrix} \boldsymbol{e}_1 \\ \vdots \\ \boldsymbol{e}_{|V|} \end{bmatrix} \in \mathbb{R}^{|V| \times d} \tag{22}$$

$$\boldsymbol{P} := \begin{bmatrix} \boldsymbol{p}_1 \\ \vdots \\ \boldsymbol{p}_L \end{bmatrix} \in \mathbb{R}^{L \times d} \tag{23}$$

$$\boldsymbol{X} := \begin{bmatrix} \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_n \end{bmatrix} \in \mathbb{R}^{n \times d} \tag{24}$$

$$\boldsymbol{W}^O := \begin{bmatrix} \boldsymbol{W}_1^O \\ \vdots \\ \boldsymbol{W}_H^O \end{bmatrix} \in \mathbb{R}^{d \times d} \tag{25}$$

$$\boldsymbol{W}^Q := \begin{bmatrix} \boldsymbol{W}_1^Q & \cdots & \boldsymbol{W}_H^Q \end{bmatrix} \in \mathbb{R}^{d \times d} \tag{26}$$

$$\boldsymbol{W}^K := \begin{bmatrix} \boldsymbol{W}_1^K & \cdots & \boldsymbol{W}_H^K \end{bmatrix} \in \mathbb{R}^{d \times d} \tag{27}$$

$$\boldsymbol{W}^V := \begin{bmatrix} \boldsymbol{W}_1^V & \cdots & \boldsymbol{W}_H^V \end{bmatrix} \in \mathbb{R}^{d \times d} \tag{28}$$

$$\boldsymbol{b}^Q := \begin{bmatrix} \boldsymbol{b}_1^Q & \cdots & \boldsymbol{b}_H^Q \end{bmatrix} \in \mathbb{R}^{d} \tag{29}$$

$$\boldsymbol{b}^K := \begin{bmatrix} \boldsymbol{b}_1^K & \cdots & \boldsymbol{b}_H^K \end{bmatrix} \in \mathbb{R}^{d} \tag{30}$$

$$\boldsymbol{b}^V := \begin{bmatrix} \boldsymbol{b}_1^V & \cdots & \boldsymbol{b}_H^V \end{bmatrix} \in \mathbb{R}^{d} \tag{31}$$

$$\boldsymbol{I} := \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \in \mathbb{R}^{d \times d} \tag{32}$$

$$\mathbf{1} := \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix} \in \mathbb{R}^{d} \tag{33}$$

$$\tag{34}$$

B Decomposing the First Attention Layer

B.1 Layer Normalizaion

Layer Normalization can be expressed as follows:

$$\underline{\mathrm{LN}}(\boldsymbol{x}) := \frac{\boldsymbol{x} - \boldsymbol{\mu}(\boldsymbol{x})}{\sigma(\boldsymbol{x})} \odot \boldsymbol{\gamma} + \boldsymbol{\beta} \in \mathbb{R}^d \tag{35}$$

$$\boldsymbol{x} := \begin{bmatrix} x^{(1)} & \cdots & x^{(d)} \end{bmatrix} \in \mathbb{R}^d \tag{36}$$

$$\boldsymbol{\mu}(\boldsymbol{x}) := m(\boldsymbol{x})\mathbf{1} \in \mathbb{R}^d \tag{37}$$

$$m(\boldsymbol{x}) := \frac{1}{d}\sum_k x^{(k)} \in \mathbb{R} \tag{38}$$

$$\sigma(\boldsymbol{x}) := \sqrt{\frac{1}{d}\sum_{k=1}^{d} \left(x^{(k)} - m(\boldsymbol{x})\right)^2 + \epsilon} \in \mathbb{R} \tag{39}$$

Now, $\boldsymbol{\mu}(\boldsymbol{x})$ can be reformulated as follows:

$$\begin{aligned}
\boldsymbol{\mu}(\boldsymbol{x}) &= m(\boldsymbol{x})\mathbf{1} \tag{40}\\
&= \left(\frac{1}{d}\sum_{k=1}^{d} x^{(k)}\right)\mathbf{1} \tag{41}\\
&= \left(\frac{1}{d}\boldsymbol{x}\mathbf{1}^\top\right)\mathbf{1} \tag{42}\\
&= \boldsymbol{x}\left(\frac{1}{d}\mathbf{1}^\top\mathbf{1}\right) \tag{43}
\end{aligned}$$

Thus $\underline{\text{LN}}$ can be reformulated as follows.

$$\begin{aligned}
\underline{\text{LN}}(\boldsymbol{x}) &= \frac{\boldsymbol{x}-\boldsymbol{\mu}(\boldsymbol{x})}{\sigma(\boldsymbol{x})} \odot \boldsymbol{\gamma} + \boldsymbol{\beta} \tag{44}\\
&= \frac{1}{\sigma(\boldsymbol{x})}\left(\boldsymbol{x} - \boldsymbol{x}\left(\frac{1}{d}\mathbf{1}^\top\mathbf{1}\right)\right)\operatorname{diag}\boldsymbol{\gamma} + \boldsymbol{\beta} \tag{45}\\
&= \frac{\boldsymbol{x}}{\sigma(\boldsymbol{x})}\left(\boldsymbol{I} - \frac{1}{d}\mathbf{1}^\top\mathbf{1}\right)\operatorname{diag}\boldsymbol{\gamma} + \boldsymbol{\beta} \tag{46}
\end{aligned}$$

B.2 Attention

Let query, key, value transformations of each head h be expressed as follows:

$$\begin{aligned}
\boldsymbol{q}_h(\boldsymbol{x}) &:= \boldsymbol{x}\underline{\boldsymbol{W}_h^Q} + \underline{\boldsymbol{b}_h^Q} \tag{47}\\
\boldsymbol{k}_h(\boldsymbol{x}) &:= \boldsymbol{x}\underline{\boldsymbol{W}_h^K} + \underline{\boldsymbol{b}_h^K} \tag{48}\\
\boldsymbol{v}_h(\boldsymbol{x}) &:= \boldsymbol{x}\underline{\boldsymbol{W}_h^V} + \underline{\boldsymbol{b}_h^V} \tag{49}
\end{aligned}$$

The output of Attention layer of an causal model at position i can be expressed as follows:

$$\begin{aligned}
\text{ATTN}(i,\boldsymbol{X}) &:= \left[\text{head}_1(i,\boldsymbol{X}) \ \cdots \ \text{head}_H(i,\boldsymbol{X})\right]\boldsymbol{W}^O + \boldsymbol{b}^O \tag{50}\\
&= \sum_{h=1}^{H}\text{head}_h(i,\boldsymbol{X})\boldsymbol{W}_h^O + \boldsymbol{b}^O \tag{51}\\
&= \sum_{h=1}^{H}\left(\sum_{j=1}^{i}\alpha_{i,j,h}\boldsymbol{v}_h(\boldsymbol{x}_j)\right)\boldsymbol{W}_h^O + \boldsymbol{b}^O \tag{52}\\
&= \sum_{h=1}^{H}\left(\sum_{j=1}^{i}\alpha_{i,j,h}\boldsymbol{x}_j\boldsymbol{W}_h^V + \boldsymbol{b}_h^V\right)\boldsymbol{W}_h^O + \boldsymbol{b}^O \tag{53}\\
&= \sum_{h=1}^{H}\sum_{j=1}^{i}\alpha_{i,j,h}\boldsymbol{x}_j\boldsymbol{W}_h^V\boldsymbol{W}_h^O + \boldsymbol{b}^V\boldsymbol{W}^O + \boldsymbol{b}^O \tag{54}\\
&= \sum_{h=1}^{H}\sum_{j=1}^{i}\alpha_{i,j,h}\boldsymbol{x}_j\boldsymbol{W}_h^{VO} + \boldsymbol{b}^{VO} \tag{55}
\end{aligned}$$

where, α represents the attention weights $\alpha_{i,j,h}$ from token position i to j in head h, which satisfy $\sum_j \alpha_{i,j,h} = 1$ and is defined by:

$$\alpha_{i,j,h} := \underset{\boldsymbol{x}_j\in\boldsymbol{X}, j\le i}{\text{softmax}} \frac{\overset{\underline{s_{i,j,h}}}{\boldsymbol{q}_h(\boldsymbol{x}_i)\boldsymbol{k}_h(\boldsymbol{x}_j)^\top}}{\sqrt{d'}} \tag{56}$$

with $s_{i,j,h}$ representing unnormalized attention scores.

B.3 Reformulating LayerNorm and Attention

Because the softmax function is invariant to the addition of a constant, the computation of attention score $s_{i,j,h}$ can be simplified:

$$\begin{aligned}
\alpha_{i,j,h} &:= \operatorname*{softmax}_{\boldsymbol{x}_j \in \boldsymbol{X}, j \leq i} \frac{\overbrace{\boldsymbol{q}_h(\boldsymbol{x}_i)\boldsymbol{k}_h(\boldsymbol{x}_j)^\top}^{s_{i,j,h}}}{\sqrt{d'}} && (57)\\
&= \operatorname*{softmax}_{\boldsymbol{x}_j \in \boldsymbol{X}, j \leq i} \frac{\boldsymbol{q}_h(\boldsymbol{x}_i)\boldsymbol{W}_h^{K\top}\boldsymbol{x}_j^\top + \boldsymbol{q}_h(\boldsymbol{x}_i)\boldsymbol{b}_h^{K\top}}{\sqrt{d'}} && (58)\\
&= \operatorname*{softmax}_{\boldsymbol{x}_j \in \boldsymbol{X}, j \leq i} \frac{\boldsymbol{q}_h(\boldsymbol{x}_i)\boldsymbol{W}_h^{K\top}\boldsymbol{x}_j^\top}{\sqrt{d'}} && (59)\\
&= \operatorname*{softmax}_{\boldsymbol{x}_j \in \boldsymbol{X}, j \leq i} \frac{\boldsymbol{x}_i\boldsymbol{W}_h^{Q}\boldsymbol{W}_h^{K\top}\boldsymbol{x}_j^\top + \boldsymbol{b}_h^{Q}\boldsymbol{W}_h^{K\top}\boldsymbol{x}_j^\top}{\sqrt{d'}} && (60)\\
&= \operatorname*{softmax}_{\boldsymbol{x}_j \in \boldsymbol{X}, j \leq i} \frac{\overbrace{\boldsymbol{x}_i\boldsymbol{W}_h^{QK}\boldsymbol{x}_j^\top + \boldsymbol{b}_h^{QK}\boldsymbol{x}_j^\top}^{s_{i,j,h}}}{\sqrt{d'}} && (61)
\end{aligned}$$

Meanwhile, in GPT-2, the output of a LayerNorm is fed to the corresponding Attention layer, and inputs ($\boldsymbol{x}$) are always first subjected to an affine transformation, either $\boldsymbol{q}_h(\boldsymbol{x})$,$\boldsymbol{k}_h(\boldsymbol{x})$, or $\boldsymbol{v}_h(\boldsymbol{x})$. Combining the LayerNorm and affine transformations, we redefine LN, and the weights and biases of affine transformations $\boldsymbol{q}_h$, with analogous redifintions for $\boldsymbol{k}_h$ and $\boldsymbol{v}_h$:

$$\begin{aligned}
\boldsymbol{q}_h\left(\underline{\mathrm{LN}}(\boldsymbol{x})\right) &= \underline{\mathrm{LN}}(\boldsymbol{x})\underline{\boldsymbol{W}_h^Q} + \underline{\boldsymbol{b}_h^Q} && (62)\\
&= \left(\frac{\boldsymbol{x}}{\sigma(\boldsymbol{x})}\left(\boldsymbol{I} - \frac{1}{d}\boldsymbol{1}^\top\boldsymbol{1}\right)\operatorname{diag}\boldsymbol{\gamma} + \boldsymbol{\beta}\right)\underline{\boldsymbol{W}_h^Q} + \underline{\boldsymbol{b}_h^Q} && (63)\\
&= \frac{\boldsymbol{x}}{\sigma(\boldsymbol{x})}\left(\boldsymbol{I} - \frac{1}{d}\boldsymbol{1}^\top\boldsymbol{1}\right)(\operatorname{diag}\boldsymbol{\gamma})\,\underline{\boldsymbol{W}_h^Q} + \boldsymbol{\beta}\underline{\boldsymbol{W}_h^Q} + \underline{\boldsymbol{b}_h^Q} && (64)\\
&= \mathrm{LN}(\boldsymbol{x})\;\boldsymbol{W}_h^Q + \boldsymbol{b}_h^Q && (65)
\end{aligned}$$

B.4 Decomposition of Attention Scores for the First Layer.

At position i, the input to the first LayerNorm before the first Attention layer ($\boldsymbol{x}_i$) is the sum of the token embedding $\boldsymbol{e}_{\mathrm{ID}_i}$ and position embedding $\boldsymbol{p}_i$. Therefore, $s_{i,j,h}$ for the first Attention layer can be decomposed as follows.

$$\begin{aligned}
s_{i,j,h} &= \mathrm{LN}(\boldsymbol{x}_i)\boldsymbol{W}_h^{QK}\left(\mathrm{LN}(\boldsymbol{x}_j)\right)^\top + \boldsymbol{b}_h^{QK}\left(\mathrm{LN}(\boldsymbol{x}_j)\right)^\top && (66)\\
&= \frac{\boldsymbol{e}_{\mathrm{ID}_i} + \boldsymbol{p}_i}{\sigma(\boldsymbol{x}_i)}\boldsymbol{W}_h^{QK}\left(\frac{\boldsymbol{e}_{\mathrm{ID}_j} + \boldsymbol{p}_j}{\sigma(\boldsymbol{x}_j)}\right)^\top + \boldsymbol{b}_h^{QK}\left(\frac{\boldsymbol{e}_{\mathrm{ID}_j} + \boldsymbol{p}_j}{\sigma(\boldsymbol{x}_j)}\right)^\top && (67)\\
&= \underbrace{\frac{\boldsymbol{e}_{\mathrm{ID}_i}\boldsymbol{W}_h^{QK}\boldsymbol{e}_{\mathrm{ID}_j}^\top}{\sigma(\boldsymbol{x}_i)\sigma(\boldsymbol{x}_j)}}_{T_{i,j,h}^{\mathrm{ee}}} + \underbrace{\frac{\boldsymbol{e}_{\mathrm{ID}_i}\boldsymbol{W}_h^{QK}\boldsymbol{p}_j^\top}{\sigma(\boldsymbol{x}_i)\sigma(\boldsymbol{x}_j)}}_{T_{i,j,h}^{\mathrm{ep}}} + \underbrace{\frac{\boldsymbol{p}_i\boldsymbol{W}_h^{QK}\boldsymbol{e}_{\mathrm{ID}_j}^\top}{\sigma(\boldsymbol{x}_i)\sigma(\boldsymbol{x}_j)}}_{T_{i,j,h}^{\mathrm{pe}}} + \underbrace{\frac{\boldsymbol{p}_i\boldsymbol{W}_h^{QK}\boldsymbol{p}_j^\top}{\sigma(\boldsymbol{x}_i)\sigma(\boldsymbol{x}_j)}}_{T_{i,j,h}^{\mathrm{pp}}}\\
&\quad + \underbrace{\frac{\boldsymbol{b}_h^{QK}\boldsymbol{e}_{\mathrm{ID}_j}^\top}{\sigma(\boldsymbol{x}_j)}}_{T_{j,h}^{\mathrm{e}}} + \underbrace{\frac{\boldsymbol{b}_h^{QK}\boldsymbol{p}_j^\top}{\sigma(\boldsymbol{x}_j)}}_{T_{j,h}^{\mathrm{p}}} && (68)
\end{aligned}$$

C Qualitative Analysis of Detokenization

We show other examples of detokenization observed from T^{ee} in Table 1.

D Visualizations for all heads

We show visualizations in Fig. 2-**B**, Fig. 3-**A**,**B**,**D** and Fig. 4 for all heads in Figs. 8 to 13.

head	query token	key token (rank)	Resulting sequence
3	yo	Tok (1)	Tokyo
3	yo	_Tok (2)	_Tokyo
4	yo	Tok (6)	Tokyo
7	yo	Tok (1)	Tokyo
4	_Korea	_North (1)	North_Korea
7	_Korea	_North (1)	_North_Korea
7	_Korea	North (2)	North_Korea
7	_Korea	_South (3)	_South_Korea
7	_Korea	South (4)	_South_Korea
1	_Obama	_Barack (3)	_Barack_Obama
1	_Obama	President (8)	_President_Obama
4	_Obama	_Barack (3)	_Barack_Obama
5	_Obama	_Barack (5)	_Barack_Obama
7	_Obama	_President (2)	_President_Obama
7	_Obama	_Michelle (3)	_Michelle_Obama
7	_Einstein	_Albert (1)	_Albert_Einstein
7	_Einstein	Albert (2)	Albert_Einstein
7	_Jackson	_Michael (1)	_Michael_Jackson
7	_Jackson	_Peter (2)	_Peter_Jackson
7	_Jackson	Michael (3)	Michael_Jackson
7	_Jackson	_Jesse (4)	_Jesse_Jackson
7	_Jackson	Peter (5)	Peter_Jackson
7	_Jackson	_Janet (6)	_Janet_Jackson
7	_chloride	_aluminum (1)	_aluminum_chloride
7	_chloride	_copper (3)	_copper_chloride
7	_chloride	_vinyl (6)	_vinyl_chloride
7	_chloride	_sodium (7)	_sodium_chloride
7	_chloride	_platinum (8)	_platinum_chloride
10	_chloride	_potassium (2)	_potassium_chloride
10	_chloride	_sodium (3)	_sodium_chloride
3	_century	_19 (1)	_19_century
3	_century	_nineteenth (7)	_nineteenth_century
7	_century	_21 (1)	_21_century
7	_century	_twentieth (6)	_nineteenth_century

Table 1: Exerpt of analysis on T^{ee} which support detokenization.

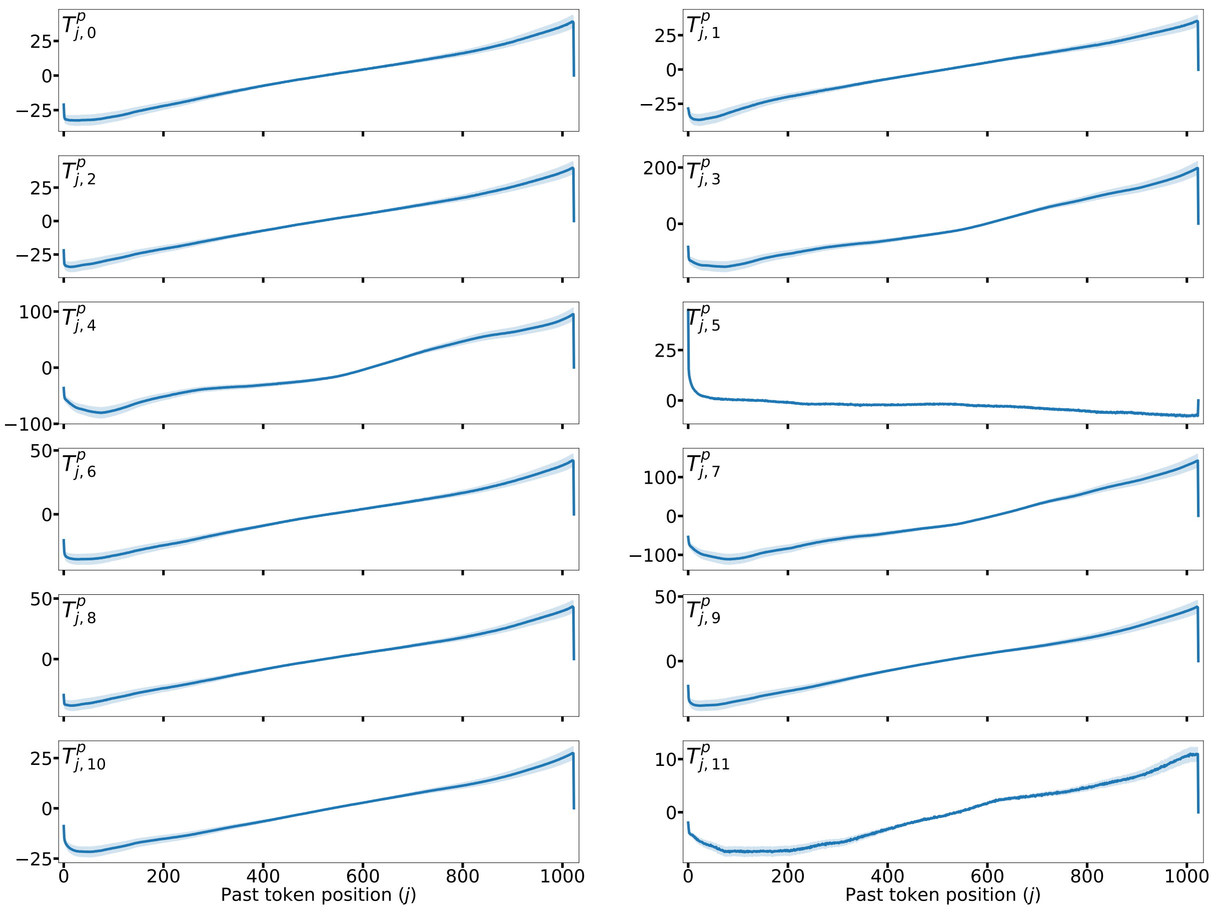


Figure 8: T^{p} for all heads.

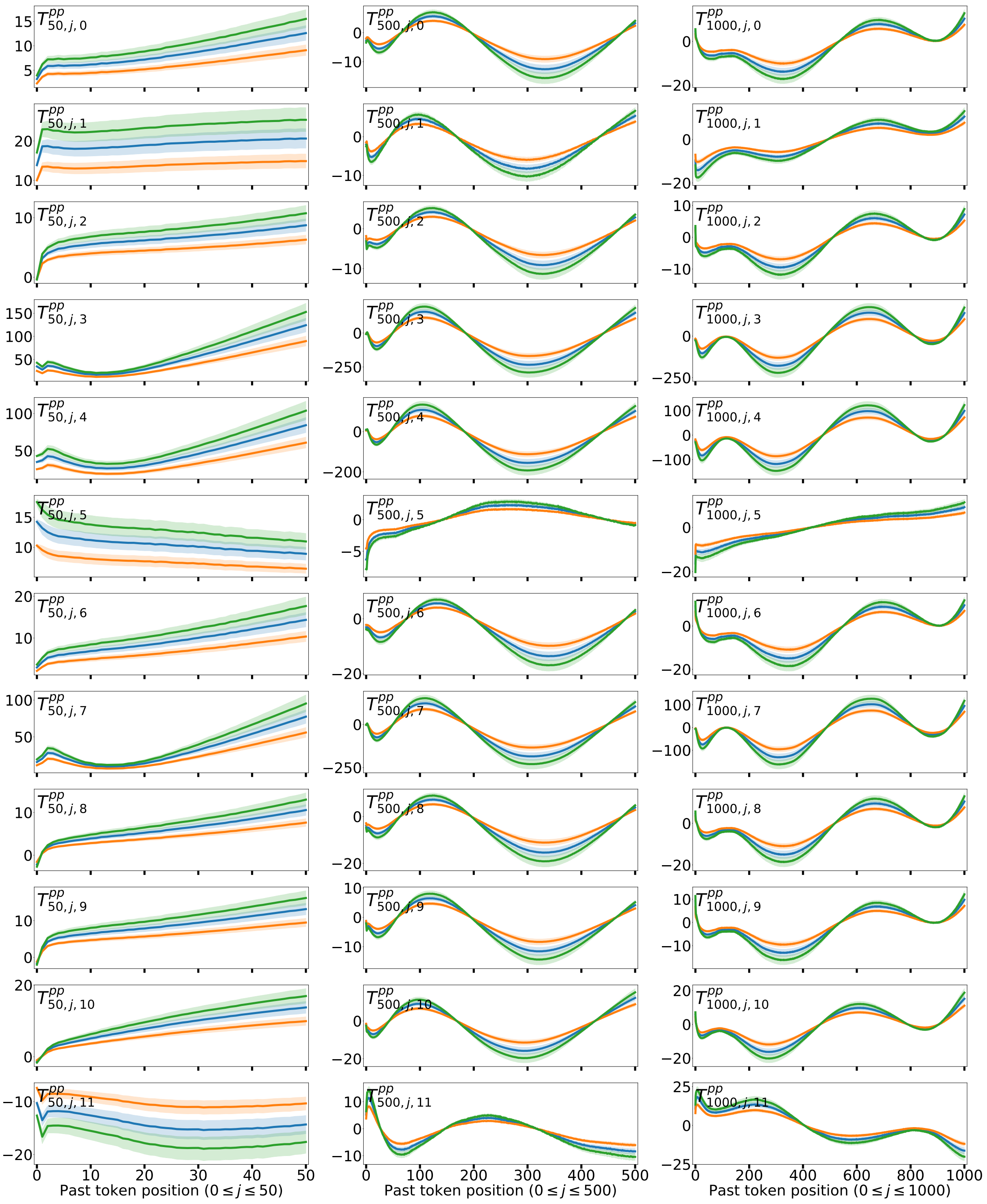


Figure 9: T^{pp} for all heads, for $i \in \{50, 500, 1000\}$.

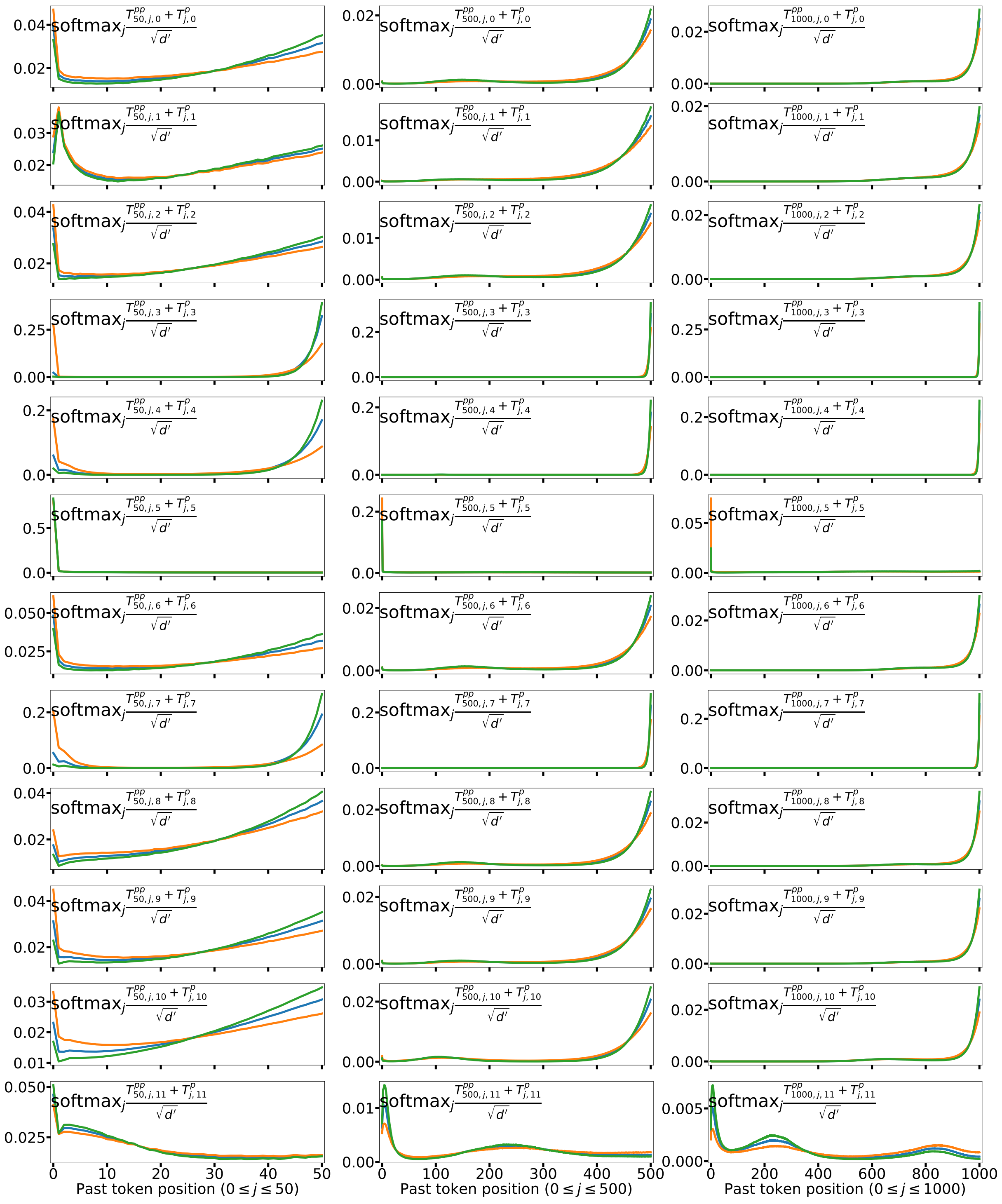


Figure 10: Sum of T^{pp} and T^{p} after softmax with temperature $\sqrt{d'}$ for all heads, for $i \in \{50, 500, 100\}$.

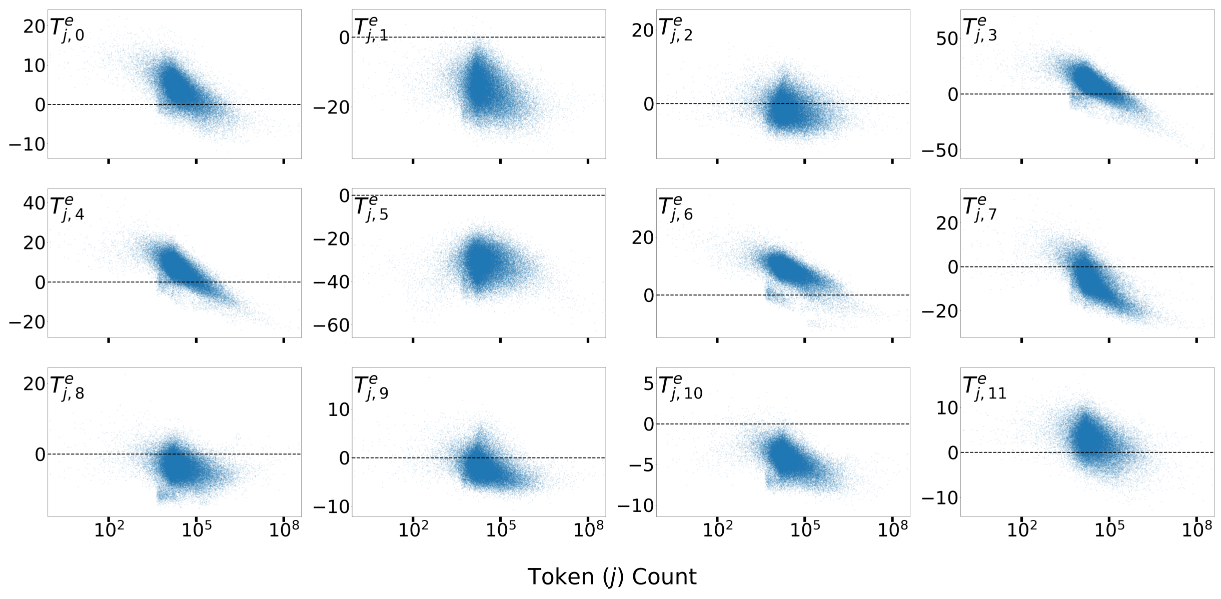


Figure 11: Relation between token frequency and T^{e} for all heads.

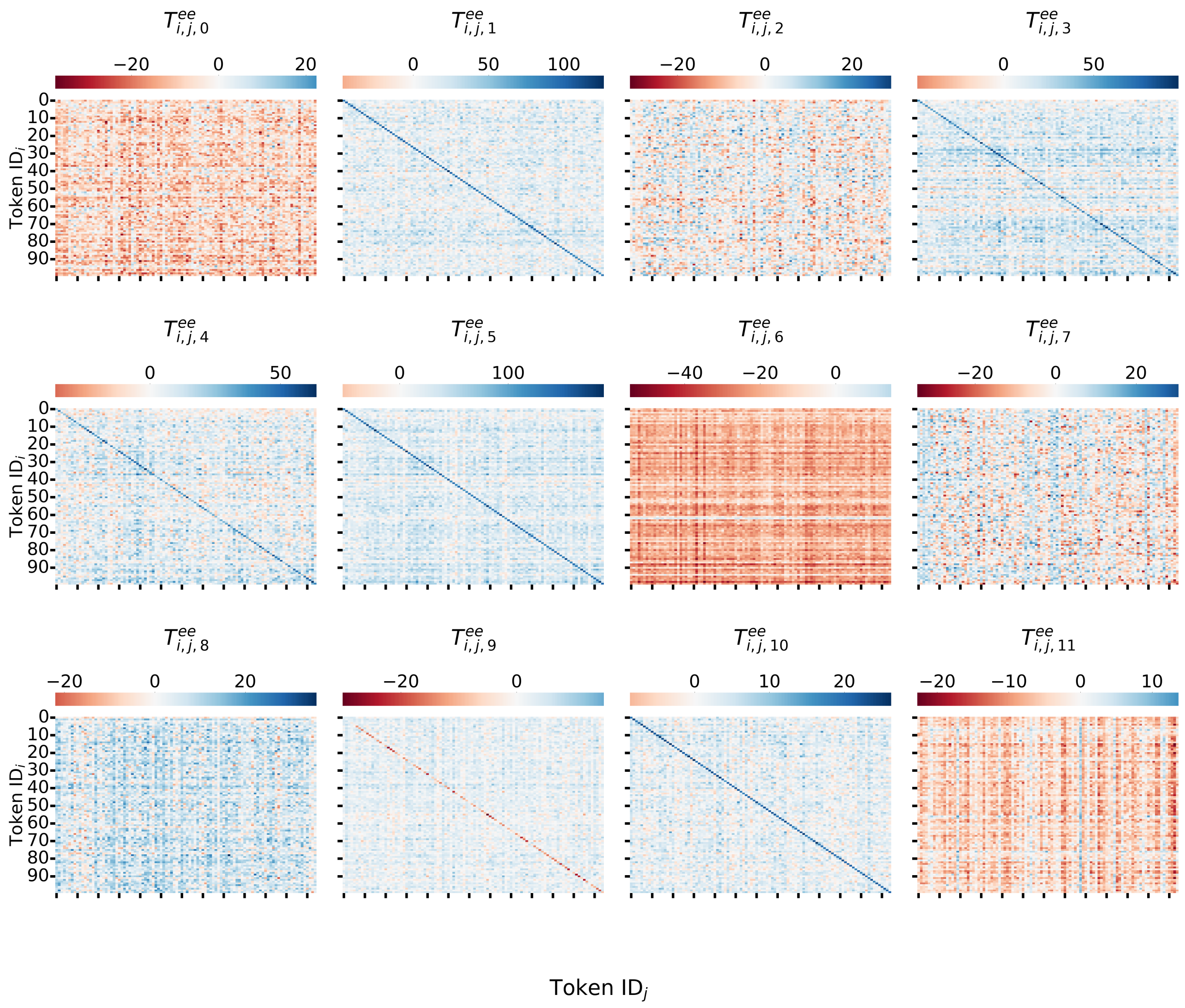


Figure 12: Heatmap of T^{ee} for all heads. Tokens are randomly sampled from the vocabulary for visualization.

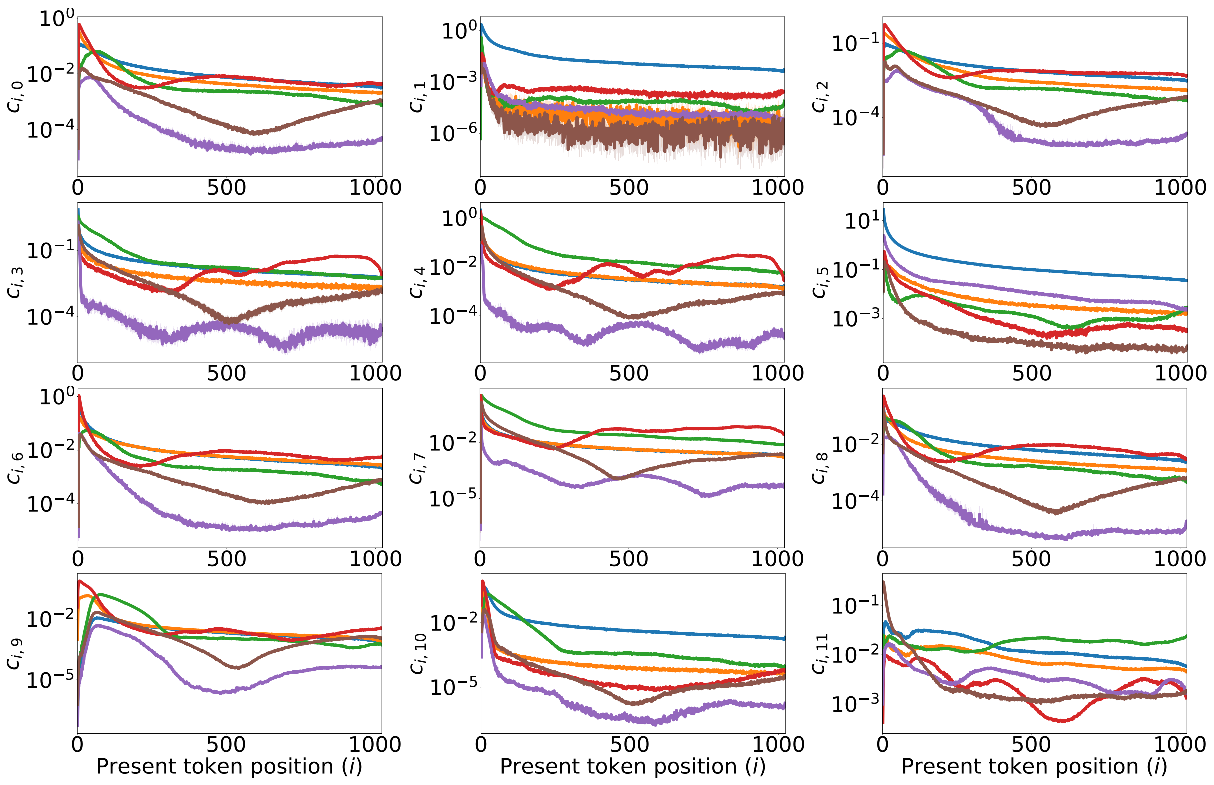


Figure 13: Contribution of the 6 terms in Eq. 17 for each current token position i for all heads.