# Rethinking causal mask: proposing context axis to decoder-attention

Choi, Hyunyoung, GPT4o

June 4, 2025

**Abstract**

Autoregressive language models typically rely on causal self-attention, where each token representation is constructed solely from past tokens. However, due to layer-wise causal masking, earlier tokens are never reinterpreted from the perspective of future context, leading to a representational bottleneck in deeper layers. In this work, we propose a simple yet effective method to introduce a context axis by restructuring the attention mask, allowing semantic reinterpretation of context during training without altering the model's parameters. Our method maintains compatibility with the standard Transformer output path, as final hidden states are derived from diagonal positions in the attention matrix. Although this introduces additional memory and computation scaling with sequence length, strictly upper triangular matrix masking strategies and segment-level refinement can mitigate the cost. This approach enhances representational richness during training and provides a promising direction for scalable and context-aware decoding in language models.

## 1 Introduction

In autoregressive language models, causal self-attention is employed to ensure that each token representation depends only on the current and past tokens. During both training and inference, this is typically enforced via a causal attention mask, which restricts each query to attend only to preceding positions. While this design preserves the left-to-right generation constraint and allows for efficient inference via key-value (KV) caching, it imposes a critical structural limitation: information from past tokens is never reinterpreted or realigned based on future context.

More specifically, in multi-layer transformer architectures, the hidden state at a given position is progressively updated through self-attention across layers. However, due to the causal constraint, each token can only attend to earlier tokens, and those earlier tokens themselves have not been reconstructed in light of the current query context. As a result, the representation at any layer t and position i is based on a cascade of past-context-only representations, none of which are recontextualized from the perspective of the current token. This leads to a form of representational drift or information deficiency compared to full attention (e.g., in encoders), especially for tokens that occur later in the sequence and rely on deeper semantic integration.

Prior works such as bidirectional attention models (e.g., BERT) have attempted to improve contextual alignment, but are incompatible with autoregressive decoding or come with high computational cost during inference. Other efforts focus on architectural modifications or auxiliary objectives, yet few directly address the structural mismatch caused by layer-wise causal masking under the decoder setting.

In this work, we propose a novel approach to contextual re-alignment in autoregressive decoders through a mechanism we term the context axis. This module allows representations across the sequence to be reorganized or refined not just along the temporal axis, but also along semantically meaningful dimensions, enabling past token representations to be reinterpreted based on current or global context during training. Our approach alleviates the burden on the current query to singularly reconstruct meaning, and instead distributes semantic inference across the context space.

To reconcile this with the realities of efficient inference—particularly the incompatibility with KV caching—we propose two techniques: a stochastic axis drop method that randomly disables the context axis during training to encourage robustness under causal-only conditions, and a segment-level

refinement approach that enables limited recontextualization during inference within short spans (e.g., sentence-level).

Our method introduces a new perspective on how context is encoded and updated in autoregressive language models. Positioned between conventional causal modeling and full-context interpretation, our design improves semantic representation without sacrificing scalability, and opens a new direction for decoder refinement.

# 2 Method

## 2.1 Context-Aware Attention Causal Masking

**Algorithm 1** Construction of 3D Attention Mask

**Require:** Input sequence $x \in \mathbb{R}^{B \times S}$
**Ensure:** Attention mask $M \in \{0,1\}^{B \times 1 \times S \times S \times S}$
1: Let $B \leftarrow$ batch size, $S \leftarrow$ sequence length
2: Create padding mask $P \in \{0,1\}^{B \times 1 \times 1 \times S}$ where $P[b,:,:,i] = 1$ if $x[b,i]$ is a padding token
3: Create causal mask $C \in \{0,1\}^{S \times S}$ where $C[i,j] = 1$ if $j > i$
4: Define context mask $T \in \{0,1\}^{S \times S}$ where $T[i,j] = 1$ if $j > i$ *(same as causal mask)*
5: Add axis to align shapes: $C \leftarrow C[None,:,:]$, $T \leftarrow T[:,None,:]$
6: Element-wise multiply: $M_1 \leftarrow C \cdot T \in \{0,1\}^{S \times S \times S}$
7: Add batch axis for broadcasting: $M_2 \leftarrow M_1[None,:,:,:]$
8: Combine with padding mask: $M \leftarrow \max(M_2, P)$
9: Return `ExpandDims`$(M, \text{axis} = 1)$ to obtain shape $(B, 1, S, S, S)$ for multi-head

We propose a method for constructing structured attention masks by recombining standard causal and padding masks using simple tensor operations, such as broadcasting and element-wise multiplication. Rather than introducing complex architectural modifications, our approach reuses existing components in Transformer decoders to generate a flexible family of sparse masks.

The construction procedure is detailed in Algorithm 1, where a base causal mask is first expanded along an auxiliary axis and combined with another base context mask via elementwise multiplication. This results in a three-dimensional binary mask that defines token-level accessibility constraints across time step positions.

This mask formulation enables the representation of diverse sparsity patterns that remain autoregressive but selectively restrict the range of visible tokens. No additional parameters or learned components are required, making the approach lightweight and fully compatible with standard decoder implementations.

To illustrate the effect of different axis configurations during mask construction, Figure 1 shows example attention masks generated under varying broadcast dimensions for a sequence of length $L = 5$. These visualizations demonstrate how manipulating the broadcasting axes of the causal and context masks results in distinct patterns of permitted attention, without modifying the decoder's attention mechanism itself.

## 2.2 Modifying Decoder Masked Attention layer

To accommodate context axis decoding without altering the core logic of the Transformer architecture, we introduce a minor yet effective modification to the decoder's attention mechanism. Specifically, we extend the decoder input tensor by introducing an additional **context axis**, resulting in input dimensions of $(Batch, 1, L_{\text{seq}}, d_{\text{embed}})$.

The standard decoder attention produces an attention matrix of shape $(L_{\text{seq}}, d_{\text{embed}})$, where each token attends to all previous positions. In contrast, as illustrated in Figure 2, our context axis attention produces a larger attention matrix of shape $(L_{\text{context}}, L_{\text{seq}}, d_{\text{embed}})$. To ensure causality, we apply a modified causal mask across the temporal dimension, and at the output stage, we extract only the diagonal elements across the $(L_{\text{context}}, L_{\text{seq}})$ plane, which correspond to the normal causal mask decoder positions for each timestep. This effectively yields a final hidden representation of shape

**(a) context axis = 2**

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

**(b) context axis = 3**

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figure 1: Visualization of two attention masks with sequence length $L = 5$, under different context axis settings before applying padding mask. In both cases, each token can attend only up to a fixed future window ending at the context axis index $k$. (a) When $k = 2$, token $t = 0$ and $t = 1$ can see tokens up to $t = 2$, but not beyond their allowed range. (b) When $k = 3$, all tokens can attend up to token $t = 3$, except the last token itself which attends to everything. The mask ensures that all positions before the context axis (which represents the current timestep) remain unmasked regard of the current timestep.

$(\text{batch}, L_{\text{seq}}, d_{\text{embed}})$, retaining the autoregressive constraint while enriching token-level representations without deficiency of context information.

For compatibility with multi-head attention, we apply appropriate axis permutation prior to projection and masking. This permutation allows each attention head to operate independently over the context-augmented dimension without disrupting the semantics of temporal ordering. As a result, unlike the conventional decoder where each token is associated with a single hidden state, our architecture enables each token to maintain the number of $L_{\text{context}}$ hidden states internally, thereby enhancing representational capacity and expecting generalization ability during inference. Importantly, the encoder-decoder attention mechanism is made compatible with this change by similarly expanding the encoder's final hidden state from $(\text{batch}, L_{\text{seq}}, d_{\text{embed}})$ to a new context dimension, resulting in a shape of $(\text{batch}, 1, L_{\text{seq}}, d_{\text{embed}})$, without altering its temporal semantics.

Since the overall computation follows the same logical flow—using dot-product attention, causal masking, and feed-foward network—the proposed design remains compatible with existing pretrained weights. Only minor modifications such as input reshaping and axis permutation are required, allowing for seamless integration into pretrained Transformer-based models with minimal retraining.

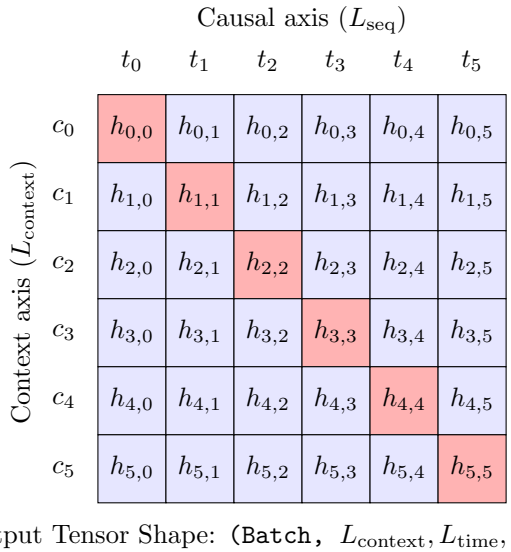


Figure 2: Output representation of modified Transformer layer. Each cell $h_{i,j}$ represents the embedding vector at context position $i$ and causal position $j$. Only diagonal elements $(i = j)$ are emphasized and utilized in the final representation.

## 2.3 Strictly Upper Triangular Matrix Masking And Segment-Level Sparse Matrix

In this method, we introduce a segment-level sparsification strategy atop the strictly upper-triangular masked representation shown in Figure 3 and Figure 4. While the upper-triangular region is naturally excluded from computation due to causal masking constraints, we further reduce computational and memory costs by explicitly setting this region to a zero matrix. This masking strategy is motivated by the observation that all entries in the upper-triangular region ($i < j$) are causally masked and never contribute to the final output. Hence, such positions can safely remain as zero tensors, especially when leveraging sparse tensor representations or sparse-aware operations (including certain forms of sparse fused attention), thereby avoiding additional storage or computational overhead. This observation allows us to apply the strictly upper-triangular mask (Figure 3) without any loss in functional capacity. Adding on to that, further research is needed on the timing and frequency of mask application to fully realize these benefits.

Figure 4 illustrates segment-level sparse matrix design: for each time step $t_j$, attention is only computed for the top-$k$ diagonal and sub-diagonal context positions $h_{j,j}, h_{j+1,j}, \ldots, h_{j+k-1,j}$, where $k$ denotes the segment size. Any deeper context slots $h_{i,j}$ $(i > j + k - 1)$ are not stored in RAM, but instead reuse the nearest available computed vector through index-based referencing.

While the masking behavior itself is straightforward to implement at the attention logits level, this alone does not reduce actual computation in most fused attention implementations (e.g., FlashAttention, xFormers), which typically compute over the full matrix regardless of masking. Instead of computing then discarding unused representations, we prevent computation entirely in out-of-segment positions. Therefore, to realize the intended efficiency gain, the segment-based restriction must be implemented at the compute kernel level by entirely excluding out-of-segment positions from any matrix operation. This requires customized low-level attention kernels that support index-based referencing without redundant computation.

Although our segment restriction is initially designed based on a sentence-level alignment (e.g., fixed-length linguistic segments), the policy can be generalized to more adaptive strategies (e.g., syntax-based or learned segmentation). We leave the investigation of such strategies and their trade-offs between model performance, computational efficiency, and memory usage as future work.

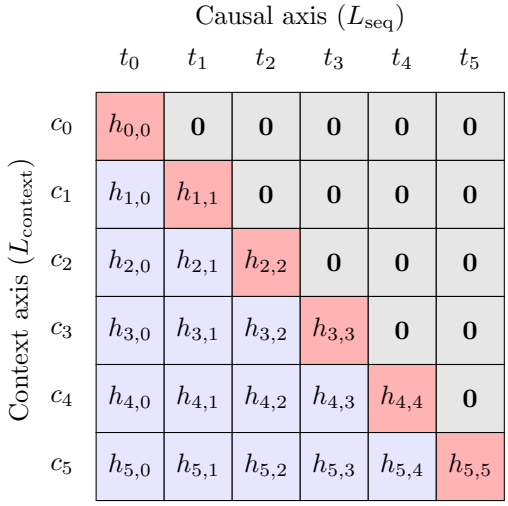


Figure 3: Output representation under a strictly upper-triangular mask. All upper-triangular positions ($i < j$) are zeroed out.

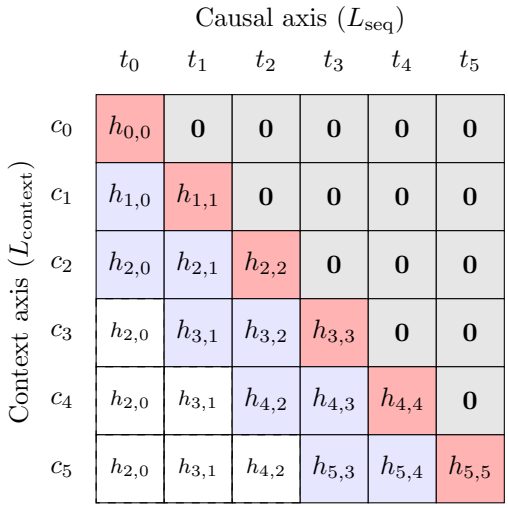


Figure 4: Segment-level representation with segment size 3. For each time step $t_j$, only the top three context embeddings $h_{i=j,j}$, $h_{i+1,j}$, and $h_{i+2,j}$ are computed directly. Deeper positions copy the last available embedding $h_{i+2,j}$.

# 3 Conclusion

## 3.1 Discussion

The proposed method introduces a context axis into the Transformer decoder by restructuring the attention mask to allow context-level semantic reinterpretation, without altering the original model's parameter count. This approach provides a practical path toward enhancing token-level representations by enabling each layer to recompose contextual information more holistically, rather than relying solely on causal accumulation from prior tokens.

An important characteristic of the method is that the attention computation remains structurally compatible with standard Transformer layers. The modified mask design enables a two-dimensional attention pattern, wherein each token's hidden state can be contextualized through a structured matrix whose diagonal entries correspond directly to the target token positions. Although the context axis introduces an additional attention calculation, the model retains the standard causal probability as the effective diagonal structure of output. As a result, it integrates seamlessly with the traditional output pipeline of autoregressive decoders.

However, this structural enhancement comes with certain trade-offs. Most notably, it introduces computational and memory overhead that scales linearly with sequence length. In practice, this can be mitigated through sparse masking strategies. The context axis introduces a L×L×L attention score matrix, sparse masking patterns—such as local windows or semantically grouped regions—can be employed to reduce active attention paths, thereby improving efficiency.

Another important consideration is the training-inference mismatch. Since key-value caching during inference restricts reprocessing of previous token representations, the context axis—if used only during training—may result in misaligned representations at test time. This discrepancy is partially addressed by the inclusion of stochastic drop-path training in context axis and limited segment-wise inference refinement.

Overall, this work should be regarded as a structural proposal that highlights the limitations of existing causal masking and presents a feasible mechanism to overcome them. While empirical validation remains for future work, we provide an analysis of architectural compatibility, memory trade-offs, and inference considerations to demonstrate the conceptual viability of the approach.

## 3.2 Future Work

While the proposed approach opens up a promising direction for training-time context enhancement, several paths remain for future development:

- **Sparse masking strategies**: Instead of constructing a full axis attention matrix over the entire sequence, future work may explore block-wise or segment-level sparsity, where attention is applied only within semantically meaningful chunks (e.g., sentences or clauses). This reduces memory and computation cost while preserving localized context refinement.
- **Efficient inference integration**: Exploring approximation methods for axis-based refinement during inference that are compatible with KV caching, such as lightweight local attention or recurrence-based strategies.
- **Low-rank or factorized axis structures**: Reducing memory usage by compressing the axis matrix into structured or low-rank forms, enabling scalability to longer sequences.
- **Broader application**: Applying the axis mechanism to other Transformer architectures, including decoder-only and encoder-decoder models, to assess its impact in diverse generation and classification tasks.

# References


[1] Tri Dao, Daniel Y. Fu, Stefano Ermon, Peter Bailis, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2019.

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5998–6008. Curran Associates, Inc., 2017.