Selective Attention: Enhancing Transformer through Principled Context Control

Xuechen Zhang
University of Michigan
zxuechen@umich.edu

Xiangyu Chang
University of California, Riverside
cxian008@ucr.edu

Mingchen Li
University of Michigan
milii@umich.edu

Amit Roy-Chowdhury
University of California, Riverside
amitrc@ece.ucr.edu

Jiasi Chen
University of Michigan
jiasi@umich.edu

Samet Oymak
University of Michigan
oymak@umich.edu

Abstract

The attention mechanism within the transformer architecture enables the model to weigh and combine tokens based on their relevance to the query. While self-attention has enjoyed major success, it notably treats all queries q in the same way by applying the mapping $V^\top \mathrm{softmax}(Kq)$, where V, K are the value and key embeddings respectively. In this work, we argue that this uniform treatment hinders the ability to control contextual sparsity and relevance. As a solution, we introduce the "*Selective Self-Attention*" (SSA) layer that augments the softmax nonlinearity with a principled temperature scaling strategy. By controlling temperature, SSA adapts the contextual sparsity of the attention map to the query embedding and its position in the context window. Through theory and experiments, we demonstrate that this alleviates attention dilution, aids the optimization process, and enhances the model's ability to control softmax spikiness of individual queries. We also incorporate temperature scaling for value embeddings and show that it boosts the model's ability to suppress irrelevant/noisy tokens. Notably, SSA is a lightweight method which introduces less than 0.5% new parameters through a weight-sharing strategy and can be fine-tuned on existing LLMs. Extensive empirical evaluations demonstrate that SSA-equipped models achieve a noticeable and consistent accuracy improvement on language modeling benchmarks.

1 Introduction

Attention is a pivotal mechanism in modern machine learning that allows the model to focus on and retrieve different parts of the data, enhancing its ability to capture contextual relationships across time and space. While it was originally developed for NLP tasks through the transformer architecture, it has enjoyed widespread success in other domains such as computer vision, sequence modeling, and reinforcement learning [45, 35, 5, 9, 39].

The canonical self-attention mechanism is a sequence-to-sequence map that outputs $\boldsymbol{X} \to \mathbb{S}(\boldsymbol{Q}\boldsymbol{K}^\top)\boldsymbol{V}$ where $\mathbb{S}(\cdot)$ denotes the row-wise softmax nonlinearity and $\boldsymbol{Q}$, $\boldsymbol{K}$, $\boldsymbol{V}$ are the query, key, and value embeddings obtained through linear projections of the input sequence $\boldsymbol{X}$. Through this process, for each query, the model creates a query-dependent composition of the input context. Importantly, the model has to accomplish two objectives: namely, capturing *semantic similarity* between tokens and also adjusting the *contextual sparsity*. Here, semantic similarity can be quantified through the angle between key-query embeddings and the contextual sparsity through the spikiness of the attention map. While the importance of the former is clear, the latter is equally important given the fact that attention maps tend to be sparse in practice [8, 43, 37, 6].

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

In this paper, we argue that these two objectives can be at odds and, as a result, the self-attention layer may struggle to achieve both objectives simultaneously due to its relatively inflexible parameterization. To address this issue, we propose the *Selective Self-Attention* (SSA) layer that aims to *decouple semantic similarity from contextual sparsity*. SSA relies on a principled application of temperature-scaling (TS) to query and value embeddings. For instance, given query embedding $\boldsymbol{q}$, rather than computing $\mathbb{S}(\boldsymbol{K}\boldsymbol{q})$, SSA computes $\mathbb{S}(\tau(\boldsymbol{q}) \cdot \boldsymbol{K}\boldsymbol{q})$ where $\tau(\boldsymbol{q})$ is the learnable inverse-temperature. Intuitively, this allows for better control of the context window because $\tau(\boldsymbol{q})$ can control *contextual sparsity* while the projection matrices $\boldsymbol{W}_k, \boldsymbol{W}_q$ can fully focus on controlling semantic similarity. Figure 1 shows an example of the learned token temperatures when training the Pythia model with SSA. In summary, we make the following theoretical and empirical contributions:

- **Query selectivity.** We prove that introducing TS to the query embeddings enhances the model's capability to express a target attention map with smaller parameter norms (Proposition 1). TS particularly helps when attention maps exhibit large variations in spikiness across different queries. Real and synthetic experiments corroborate that TS enables spikier/sharper attention maps and mitigates attention dilution. See Figure 3 as an illustration.
- **Value selectivity.** We formalize the benefit of TS on value embeddings through a denoising perspective. Namely, we describe a denoising task where the linear value projection fails to filter the noisy tokens, and demonstrate how nonlinear scaling can boost denoising capability.
- **Positional temperature.** We incorporate a term that adjusts the query-temperature according to the position in the context window. We show that this term can mitigate the dilution of attention scores caused by the increasing context length.
- **Modularity and parameter-efficiency of SSA.** Selective Attention is accomplished by introducing a parameter-efficient temperature module that can be easily integrated into existing attention models. In practice, this introduces 5% additional parameters to the model. We also introduce a weight sharing strategy that reduces the number of parameter overhead to less than 0.5% while maintaining the benefits of SSA. We reuse the attention weights within the temperature module, which results in negligible inference/latency overhead since no additional matrix multiplication is required. These methods only involve vector dot-products (at the output layer of the temperature module) and elementwise scaling of matrices.
- **Empirical benefits.** Our evaluations on the NLP benchmarks of Wikitext [27], Lambada [32], Piqa [4], Hella [51], Winogrande [38], Arc-E, and Arc-C [10] demonstrate that Selective Attention noticeably improves language modeling performance. These benefits are consistent across various models including GPT-2 [34], Pythia [3], Llama [44] and Llama3 [16], as well as during both fine-tuning and pre-training, as shown in Table 3. Additionally, evaluations on the passkey retrieval task [33, 29] reveal that SSA substantially enhances the retrieval capabilities of the transformer, shown in Table 4.

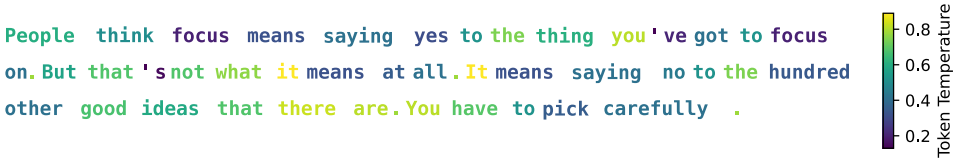


Figure 1: A quotation by Steve Jobs. We highlight tokens according to their temperatures learned by the SSA layer. Darker colors correspond to lower temperatures and receive a sparser attention map.

2 Related Work

Temperature Scaling (TS): TS is a fundamental method for controlling model behavior, influencing aspects such as stochasticity of generative LLMs, calibration and uncertainty, and imbalanced data, as highlighted in several studies [26, 22, 52]. Related to us, previous research [33, 49, 7] has also proposed utilizing a temperature term in the softmax function to enhance the *length extrapolation* capabilities of transformers. For instance, Yarn [33] scales the attention logits as a function of the sequence length and shows that this improves the perplexity when extending the context window. Our work provides a formal justification for the temperature scaling rule proposed in Yarn (see Proposition 2) and also highlights the value of adapting temperature to the individual positions. Importantly, our approach is differentiable and obviates the need for grid search required by prior works. Since we

don't focus on length generalization, we have found that position-aware temperature has a much smaller benefit compared to token-aware temperature, which is our primary contribution.

Gating mechanisms and selectivity: Various strategies have been developed to mitigate the impact of uninformative inputs in model training and processing. Gating mechanisms, originally introduced through LSTMs [19], have been proposed to selectively filter or scale down the input sequence [48, 13, 14, 25, 40]. Very recent sequence models such as Mamba (a.k.a. selective state-space model) and Griffin also incorporate gating to boost language modeling [18, 46, 54, 14, 21]. These models leverage input-dependent gating to ensure parallellizable training and enjoyed noticeable success. These methodologies inspired our approach, which incorporates TS to augment the selection capabilities of the attention layer. Specifically, TS can be viewed as an instance of gating that selectively passes or suppresses tokens to provide better control of contextual sparsity and relevance. In this light, our work also provides a mechanistic understanding of how gating mechanism can aid self-attention to improve its expressive capabilities. Finally, we highlight the concurrent work [50] which utilizes a *differential softmax parameterization* to promote spiky attention maps.

Mechanistic understanding of transformers: The importance of transformer-based models led to many research efforts on developing a stronger understanding of various aspects of transformer and attention [30, 47, 15]. While it is impossible to cover all of these works, it is evident that capability to select relevant features and promote contextual sparsity is crucial for the ability of language models to perform complex tasks such as reasoning [23, 1, 43, 53]. These have provided inspiration for us to pursue an enhanced modeling of attention's spikiness (e.g. as in Figure 3). The experiments in Figure 3 are inspired by the recent work [20] which characterizes the learnability of a ground-truth attention model via the next-token prediction objective in terms of the associated Markov transition matrix.

3 Methodology: Selective Attention Layer

Let us recap the self-attention mechanism in Transformer [45]. Canonical softmax attention admits an input sequence $\boldsymbol{X} = [\boldsymbol{x}_1 \ \dots \ \boldsymbol{x}_L]^\top \in \mathbb{R}^{L\times d}$ of length L with embedding dimension d. We then project $\boldsymbol{X}$ to obtain key, query, and value embeddings ($\boldsymbol{K} = \boldsymbol{X}\boldsymbol{W}_k,\ \boldsymbol{Q} = \boldsymbol{X}\boldsymbol{W}_q,\ \boldsymbol{V} = \boldsymbol{X}\boldsymbol{W}_v$) and compute the output of the dot-product attention as $Att(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = \mathbb{S}(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d}})\boldsymbol{V}$. Here $\mathbb{S}(\cdot) : \mathbb{R}^L \to \mathbb{R}^L_+$ denotes the softmax nonlinearity that applies row-wise and $\boldsymbol{W}_q, \boldsymbol{W}_k, \boldsymbol{W}_v \in \mathbb{R}^{d\times d}$ are learnable weight matrices. In this paper, we mainly focus on casual language modeling where each token can only attend to previous tokens in the input.

The uniform treatment of all tokens through the same softmax map could hinder the ability to control contextual sparsity and relevance. For instance, it has been observed that current Transformer language models suffer from an attention dilution issue: the longer the input sequence, the flatter the attention distribution [49, 7]. A natural solution to the dispersed attention issue is to sharpen the self-attention distribution. Selective Attention aims to provide a general strategy to control spikiness of the softmax adaptive to the query and value embedding, as well as the position of the token.

Definition 1 (Selective Self-Attention (SSA))**.** *Let $\boldsymbol{X} = [\boldsymbol{x}_1 \ \dots \ \boldsymbol{x}_L]^\top \in \mathbb{R}^{L\times d}$ be an input sequence. Let $\tau_{k/q/v}(\cdot) : \mathbb{R}^d \to \mathbb{R}^d$ be the inverse-temperature functions for keys, queries, and values, respectively. Then the embeddings for keys ($\boldsymbol{K}$), queries ($\boldsymbol{Q}$), and values ($\boldsymbol{V}$) are computed as follows:*

$$\boldsymbol{K} = \tau_k(\boldsymbol{X}) \odot \boldsymbol{X}\boldsymbol{W}_k,\ \boldsymbol{Q} = \tau_q(\boldsymbol{X}) \odot \boldsymbol{X}\boldsymbol{W}_q,\ \boldsymbol{V} = \tau_v(\boldsymbol{X}) \odot \boldsymbol{X}\boldsymbol{W}_v.$$

where $\odot$ denotes the elementwise product that assigns temperature to individual tokens. Selective Self-Attention (SSA) is then computed as $\mathbb{S}(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d}})\boldsymbol{V}$.

In essence, SSA incorporates a temperature modulation mechanism into the attention framework to enhance selectivity and context control. The inverse-temperature function $\tau(\cdot)$ is data-dependent, allowing for dynamic adjustment of attention across different parts of the input sequence. In practice, we choose $\tau_{k/q/v}$ to be a scalar valued function as vector-valued temperature does not provide a significant advantage. It is also worth mentioning that we don't restrict $\tau_{k/q/v}$ to be non-negative. As a result, our temperature scaling strategy can be seen as an application of *scalar gating* on K/Q/V embeddings, and hence, the SSA layer could also be referred to as *Scalar-Gated Attention (SGA)* layer. The GitHub repo containing SSA implementation is provided in https://github.com/umich-sota/selective_attention. Below, we discuss the design choices underlying SSA.

• **Temperature scaling for query and value tokens.** In an attention mechanism, the concepts of keys ($\boldsymbol{K}$), queries ($\boldsymbol{Q}$), and values ($\boldsymbol{V}$) play distinct roles in determining how information is weighted and combined across a sequence. Temperature functions can be applied to all of those components, designated as Key-temperature $\tau_k(\cdot)$, Query-temperature $\tau_q(\cdot)$ and Value-temperature $\tau_v(\cdot)$. We explore the advantages of each temperature function in in Appendix B.1. In practice, we employ Query-temperature $\tau_q(\cdot)$ and Value-temperature $\tau_v(\cdot)$ but don't touch the original key embeddings. The query-temperature τ_q adjusts the spikiness of the attention map associated with the query according to its embedding and position in the context window. The value-temperature τ_v enhances the model's ability to suppress irrelevant or noisy tokens, ensuring a refined aggregation of context window. In Section 4, we provide insights into theoretical and empirical benefits of incorporating these terms.While we keep the keys unmodified, guided by the intuition from word embeddings of [28] suggests that the similarity between a (key, query) pair should align with their cosine similarity. That is, $cos(key_1, query) > cos(key_2, query)$ should ideally imply that the *query* attends more to key_1 compared to key_2. Assigning temperature/gating to scale the query vector does not change this order. However, if we assign distinct scalings to key_1 and key_2, we will end up with scenarios where attention scores are flipped i.e. $\tau_1 * key_1^\top query < \tau_2 * key_2^\top query$. In other words, our intuition is that assigning gating on keys will end up influencing their relative semantic similarities to queries (which could perhaps be better achieved via attention weights). This is in contrast to query-scaling which helps decouple the semantic similarity and contextual sparsity and the associated theoretical benefits (Section 4.1 and Proposition 1).

• **Token-aware and position-aware temperature scaling.** The data-dependent inverse-temperature function is composed of two distinct components $\tau(\boldsymbol{x}) = \tau^{tok}(\boldsymbol{x}) + \tau^{pos}(\boldsymbol{x})$, $\boldsymbol{x}$ is a token within the sequence $\boldsymbol{X}$: Token-aware Temperature Scaling $\tau^{tok}(\cdot)$ and Position-aware Temperature Scaling $\tau^{pos}(\cdot)$. Token-aware Temperature Scaling $\tau^{tok}(\cdot)$ is devised to modulate the influence of individual tokens within the sequence. The formula for this component is given by $\tau^{tok}(\boldsymbol{x}) = tanh(f(\boldsymbol{x}))$, where $f(\cdot)$ represents a trainable function that adjusts the impact of the token $\boldsymbol{x}$. The activation function $tanh(\cdot)$ is used to enable the scaling function to output both positive and negative temperatures; for instance, if we want to have the option to fully-suppress a token $\tau^{tok}(\boldsymbol{x})$ can attain ≈ 0. To address the issue of dispersed attention, where increasing length of the input sequence leads to a flatter attention distribution, we introduce Position-aware Temperature Scaling. This is defined by $\tau^{pos}(\boldsymbol{x}) = 1 + \sigma(\alpha)log(n)$, where n denotes the position of the token $\boldsymbol{x}$ within the sequence $\boldsymbol{X} = [\boldsymbol{x}_1 \ \ldots \ \boldsymbol{x}_L]^\top \in \mathbb{R}^{L\times d}$, $n \in [L]$. We remark that n reflects the token length when computing the temperature of token $\boldsymbol{x}_n$, aligning with our focus on causal attention where each token is restricted to attending only to previous tokens in the sequence. α is a parameter designed to modify the scale of the factor. The non-linearity $\sigma(\cdot)$ is the sigmoid function, employed to control the range of τ^{pos} and ensure the stability of the training process.

• **Weight sharing.** We introduce a weight sharing strategy to reduce the number of parameter overhead below 0.5% (10x fewer) while maintaining the benefits of SSA. Specifically, the Position-aware Temperature Scaling term, $\tau^{pos}(\boldsymbol{x})$ only includes a single parameter α, whereas the Token-aware Temperature Scaling term $\tau^{tok}(\boldsymbol{x}) = tanh(f(\boldsymbol{x}))$, relies on a trainable function $f(\cdot)$ defined as $\boldsymbol{W}_{tmp}\text{GeLU}(\boldsymbol{W}'_{tmp}\boldsymbol{x})$, involves separate trainable parameters $\mathbf{W}_{tmp}$ and $\mathbf{W}'_{tmp}$, which increases parameter load. To improve efficiency, we (re)use the attention weights $\boldsymbol{W}_{k/q/v}$ for the temperature module by setting $f(\boldsymbol{x}) = \boldsymbol{W}_{tmp}\text{GeLU}(\boldsymbol{W}_{k/q/v}\boldsymbol{x})$. Here, SSA only adds the output layer of the MLP, a vector with few parameters. The approach only stores 3 vectors (not matrices) per attention head. This also have negligible inference/latency overhead because we don't require additional matrix multiplication. These methods only require vector dot-products (at the output layer of the temperature module) and elementwise scaling of matrices. Other strategies can also be deployed to reduce the computational overhead. We describe feature-based approach which use simple token-level statistics, such as their frequencies in training corpus. Only constant parameters per head need to be stored that reduce the number of parameter overhead below 0.1%. The deatils are shown in B.2.

Finally, we discuss conceptual connections to *sparse attention* methods in Appendix D.

4 Theoretical Insights into Selective Attention

Selective attention computes the query temperature based on the embedding and the position of the query. It also computes the value temperature based on the value embedding. In what follows, we discuss how these three components provably enhance expressivity of the attention mechanism.

4.1 The benefits of incorporating query embedding

Decoupling semantics from specificity. Consider two words: "Hinton" and "Scientist". The former is a specific instance of the latter. As a result, while we expect token embeddings of these two words to have high cosine similarity, they might benefit from different attention maps. Specifically, "Hinton" refers to a specific person and we expect it to have a more targeted attention to the context associated with it. We argue that query-temperature can aid optimization by retaining semantic similarity while allowing for distinct *specificity*. More formally, by specificity we are referring to the contextual sparsity level of a query. Denoting the combined key-query weights to be $\boldsymbol{W} = \boldsymbol{W}_q \boldsymbol{W}_k^\top$ as a problem-agnostic measure of specificity, we will consider the magnitude of the query embedding. That is, given query token $\boldsymbol{q}$, define $\text{spec}_{\boldsymbol{W}}(\boldsymbol{q}) := \|\boldsymbol{W}^\top \boldsymbol{q}\|_2$. It is well-established [43] that in order for attention map to be more sparse (hence higher specificity), the norm of the query embedding, or more generally the operator norm of $\boldsymbol{W}$, has to grow larger, justifying this definition. The following Lemma shows that, without TS, the attention weights within softmax have to be lower bounded by the ratio of *specificity difference* to *semantic distance*.

Lemma 1. *Let $\boldsymbol{W} = \boldsymbol{W}_q \boldsymbol{W}_k^\top \in \mathbb{R}^{d\times d}$ be the combined query-key matrix. Let $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^d$ be unit norm token embeddings associated with the* specific *and* general *token respectively. Suppose we wish to achieve specificities $\text{spec}_{\boldsymbol{W}}(\boldsymbol{a}) \geq L_a$ and $\text{spec}_{\boldsymbol{W}}(\boldsymbol{b}) \leq L_b$. Then, the associated $\boldsymbol{W}$ obeys $\|\boldsymbol{W}\| \geq \frac{L_a - L_b}{\|\boldsymbol{a}-\boldsymbol{b}\|_2}$.*

Above $L_a - L_b$ is the specificity_difference whereas $\|\boldsymbol{a} - \boldsymbol{b}\|_2$ is the semantic distance. The proof follows from the triangle inequality $\|\boldsymbol{W}\| \geq \frac{\|\boldsymbol{W}^\top(\boldsymbol{a}-\boldsymbol{b})\|_2}{\|\boldsymbol{a}-\boldsymbol{b}\|_2} \geq \frac{\|\boldsymbol{W}^\top \boldsymbol{a}\|_2 - \|\boldsymbol{W}^\top \boldsymbol{b}\|_2}{\|\boldsymbol{a}-\boldsymbol{b}\|_2} \geq \frac{L_a - L_b}{\|\boldsymbol{a}-\boldsymbol{b}\|_2}$.

Comparison to Selective Attention. In SSA, the effective attention weight matrix for a query $\boldsymbol{q}$ is $\boldsymbol{W} = \tau(\boldsymbol{q}) \cdot \boldsymbol{W}_q \boldsymbol{W}_k^\top$. To achieve the same specificity in Lemma 1 with SSA, we can set the temperatures as $\tau(\boldsymbol{a}) = L_a$, $\tau(\boldsymbol{b}) = L_b$, and KQ-weights as $\|\boldsymbol{W}\| = 1$ (e.g. via $\boldsymbol{W} = \boldsymbol{I}_d$). This achieves the desired specificities while maintaining that *effective weights* are upper bounded as $\|\boldsymbol{W}_{\boldsymbol{a}}\|, \|\boldsymbol{W}_{\boldsymbol{b}}\| \leq \max(L_a, L_b)$. In other words, the required norm growth is entirely decoupled from the semantic distance between the queries.

In essence, this highlights that without query-selectivity, the model weights have to grow excessively to assign different specificity to similar words. In practice, this is expected to create performance bottlenecks: (1) As the weights grow, optimization may slow down along certain directions due to vanishing softmax derivative and, (2) even if the optimization is successful, the final model could overfit or be overly sensitive to small perturbations in the context, hindering test accuracy.

This is also verified by our experiments. To study the norm growth of attention weights, we train Pythia from scratch, trainig with the SlimPajama dataset [41](our pre-training setting) and evaluate on Wikitext dataset. We examine the average norm of combined query-key matrix weight $\|\boldsymbol{W}\|$ from the average of all layers within the model. Additionally, we quantify the spikiness of the attention map computed as the ratio of the l_1–norm to the squared l_2–norm and normalized by the length, defined as $\frac{\|s\|_1}{\|s\|^2 L}$, $\boldsymbol{s}$ where $\boldsymbol{s}$ is the softmax probability vector. It takes values from 0 to 1. A smaller value indicates a sparser vector. We compute the average of the first 1000 tokens of the Wikitext dataset. The results shown in Figure 2 align with the theory. The attention weights for selective attention are smaller than the original ones, while the attention is sparser.

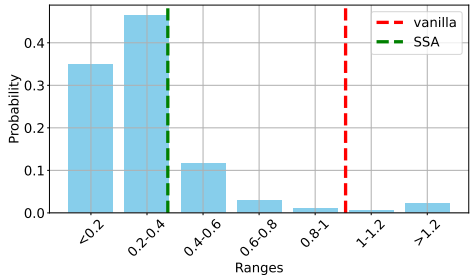
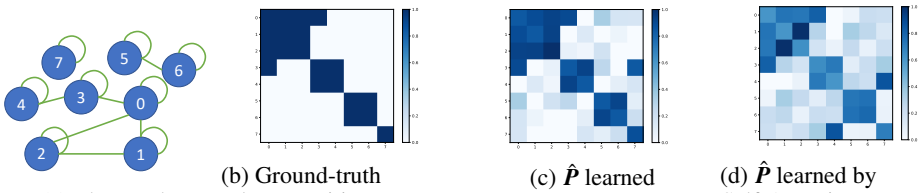


Figure 2: The operator norm of $\boldsymbol{W}$ with and without Query-temperature scaling, scaled by $\times 10^3$. The figure depicts the distribution across 1000 tokens. The dashed line is the average norm. Notably, the norm of the vanilla attention layer is approximately three times larger than that of SSA(dashed red line compare to green line). Furthermore, the vanilla attention layer exhibits a lower spikiness score (0.39) compared to SSA (0.26), where a lower value indicates higher spikiness.

Expressivity benefits of query-selectivity. A closely related consideration is whether query-selectivity can enhance expressivity. We expect that through query-temperature, the same attention head will have an easier time expressing sparse and dense attention maps associated with distinct queries. To formalize this, we investigate the ability of a single (selective) attention head to express a target attention map between



(a) The graph. (b) Ground-truth token transitions $\boldsymbol{P}_\star$ (c) $\hat{\boldsymbol{P}}$ learned by SSA (d) $\hat{\boldsymbol{P}}$ learned by Self-Attention

Figure 3: We compare 1-layer SSA and 1-layer attention when solving next-token prediction on a small vocabulary of size 8. (a) is the graph associated to the token transition dynamics. (b) is the the pairwise token transition matrix of this vocabulary. Each row of $\boldsymbol{P}_\star$ represents an attention map where a particular token is the query and all tokens in the vocabulary serve as keys (see Sec 4.1 for details). The transition matrix $\hat{\boldsymbol{P}}$ estimated by SSA in (c) is sharper and more closely resembles the optimal $\boldsymbol{P}_\star$. SSA achieves a smaller cross-entropy loss compared to vanilla attention, 0.009 vs 0.0126. The ℓ_1 approximation error of the attention map of SSA is also smaller than that of vanilla attention, 0.358 vs 0.543.

all tokens in a discrete vocabulary. Let $\mathcal{V} = \boldsymbol{e}_i{}_{i=1}^{K}$ be a vocabulary of K tokens. To capture all K^2 pairwise interactions of these tokens, we first form the sequence $\boldsymbol{E} = [\boldsymbol{e}_1 \ \dots \ \boldsymbol{e}_K]^\top \in \mathbb{R}^{K\times d}$ where each token appears uniquely and then study K attention maps associated with individual queries, i.e., $\texttt{att}(\boldsymbol{E}, \boldsymbol{e}_i)$ for $1 \le i \le K$. Stacking these together as rows, we study the $K \times K$ attention matrix $\texttt{att}(\boldsymbol{E})$. For standard attention with weights $\boldsymbol{W}$, this is given by $\texttt{att}(\boldsymbol{E}, \boldsymbol{W}) = \mathbb{S}(\boldsymbol{E}\boldsymbol{W}\boldsymbol{E}^\top)$, whereas for query-selective attention, $\texttt{att}(\boldsymbol{E}, \boldsymbol{W}) = \mathbb{S}(\tau(\boldsymbol{E}) \odot \boldsymbol{E}\boldsymbol{W}\boldsymbol{E}^\top)$.

Thanks to the softmax nonlinearity, $\texttt{att}(\boldsymbol{E})$ is a stochastic matrix where rows add up to 1. This matrix can be viewed as a Markov chain transition between different tokens, which motivates a fundamental question: *Can query-selective attention help express a larger class of stochastic matrices?* Intuitively, we expect that if a stochastic matrix $\boldsymbol{P}_\star$, which we wish to express via $\texttt{att}(\boldsymbol{E})$, exhibits a lot of spikiness variation across its rows (i.e., different queries), selectivity can better capture these.

This can be verified with a token generation experiments. Recall that we expect "bacteria" to attend to more words compared to "salmonella". We might expect more general words to have a larger number of neighbors in a graph. Accordingly, we abstract the vocabulary, which comprises words with various levels of specificity, into a simple undirected graph. This is depicted in Section 4.1. Additionally, the stochastic matrix $\boldsymbol{P}_\star$ can be derived from this graph, with the results displayed in Figure 3(a). To build the estimation of the stochastic matrix $\boldsymbol{P}_\star$ training, we conduct next token prediction experiments.

Token generation setting: Let $X \in \mathcal{V}^L$ be a sequence of length L drawn from $\mathcal{V}$. Suppose X ends with $q := x_L$. The token $Y = x_{L+1}$ that follows X will be drawn uniformly from q or one of the neighbors of q. This neighborhood is parameterized via the latent attention map $\boldsymbol{P}_\star$ which will govern the generation process. Let $\boldsymbol{E} = [\boldsymbol{e}_1 \ \dots \ \boldsymbol{e}_N]^\top$ be the token embeddings associated with the vocabulary $\mathcal{V}$. Assume elements of $\boldsymbol{E}$ have unit ℓ_2 norm. In data generation, we simple sample input sequences containing each token in the vocabulary precisely once, and sample the next token according to the attention map $\boldsymbol{P}_\star$, that is, the row of $\boldsymbol{P}_\star$ that corresponds to the final query token. We then fit a one-layer self-attention or SSA model $f(\boldsymbol{X})$ to approximate this latent dynamics. Concretely, we predict the next token $\hat{Y}$ of $f(\boldsymbol{X})$ according to the distribution $g(\boldsymbol{X}) = \mathbb{S}(\boldsymbol{C}f(\boldsymbol{X})) \in \mathbb{R}^N$. Here $\boldsymbol{C} \in \mathbb{R}^{N\times d}$ is the linear prediction head. As loss measure on how well we fit to the latent $\boldsymbol{P}_\star$ dynamics, use the cross entropy distance between $g(\boldsymbol{X})$ and the true label Y. Through this, we wish to formalize and visualize the intuitions on why "salmonella" deserves a lower temperature than "bacteria". Further experimental details are described in Appendix A.

In our experiments, besides smaller cross-entropy loss, we find that Selective Attention achieves a better approximation of $\boldsymbol{P}_\star$ as shown in Figure 3. To evaluate the similarity between the attention map $\boldsymbol{P}_\star$ and $\hat{\boldsymbol{P}}$, we also define the ℓ_1 distance between the attention maps, namely,

$$\texttt{err_map} = \|\hat{\boldsymbol{P}} - \boldsymbol{P}_\star\|_1.$$

We find that the $\texttt{err_map}_{\text{SSA}}$ is also much lower than $\texttt{err_map}_{\text{vanilla}}$ (0.358 vs 0.543). Additionally, SSA naturally assigns lower temperatures to tokens with fewer neighbors. This is in line with our expectations as fewer neighbors imply a sparser attention map. The results are shown in Table 1.

Table 1: Temperature for each depth. Nodes with the same # of neighbors share the same temperature.

# of neighbors(including itself)	1	2	3	4
nodes index	7	4,5,6	1,2,3	0
temperature	0.002	0.019	0.152	0.751

To further formalize this, we revisit Lemma 1 in terms of softmax map. Let $K = 2$ and $\boldsymbol{P}_\star = \begin{bmatrix} 1-\gamma & \gamma \\ 0 & 1 \end{bmatrix}$ be the target pairwise attention map. Here second token is highly specific (only selects itself) whereas the first token is less specific when $0 < \gamma < 1$. The following proposition establishes a variation of Lemma 1 when approximating $\boldsymbol{P}_\star$.

Proposition 1. *Suppose the embeddings $\boldsymbol{e}_1, \boldsymbol{e}_2$ have unit ℓ_2 norm with correlation $\rho = \boldsymbol{e}_1^\top \boldsymbol{e}_2$. Fix $0 < \varepsilon \le \frac{1}{2}\min(\gamma, 1-\gamma)$ and $\Gamma = \left|\log\left(\frac{1-\gamma}{\gamma}\right)\right|$. For any $\boldsymbol{W}$ obeying $\|\boldsymbol{P}_\star - \mathbb{S}(\boldsymbol{E}\boldsymbol{W}\boldsymbol{E}^\top)\|_\infty \le \varepsilon$, we have that $\|\boldsymbol{W}\| \ge \frac{\|\boldsymbol{e}_1 - \boldsymbol{e}_2\|^{-1}}{\sqrt{2-2\rho^2}}\left(\log\left(\frac{1}{4\varepsilon}\right) - \Gamma\right)$. Conversely, Selective Attention can achieve this ε-approximation with weights bounded as $\tau(\boldsymbol{e}_{1,2}) \cdot \|\boldsymbol{W}\| \le \|\boldsymbol{e}_1 - \boldsymbol{e}_2\|^{-1} \max\left(\log\left(\frac{1}{\varepsilon}\right), \frac{\Gamma}{\sqrt{1-\rho^2}}\right)$.*

4.2 The benefits of incorporating query position

The need for position-dependent scaling arises from the fact that, for a fixed weight matrix $\boldsymbol{W} = \boldsymbol{W}_q \boldsymbol{W}_k^\top$, the attention scores $\boldsymbol{s}^L = \mathbb{S}(\boldsymbol{X}\boldsymbol{W}^\top \boldsymbol{x}_L)$ become diluted as sequence length L grows. Specifically, for retrieval-type tasks, the model may want to concentrate softmax scores $\boldsymbol{s}^L$ on a single token. However, assuming unit norm tokens, the top probability in $\boldsymbol{s}^L$ is upper bounded via $\|\boldsymbol{s}^L\|_{\ell_\infty} \le \frac{1}{1+(L-1)e^{-2\|\boldsymbol{W}\|}}$. This implies that, to enforce $\|\boldsymbol{s}^L\|_{\ell_\infty}$ to be constant, we require the spectral norm lower growth rate of $\|\boldsymbol{W}\| \ge 0.5 \log L + O(1)$. This motivates our logarithmic scaling strategy which was also proposed by [33, 7].

Here we provide a more formal justification on the optimal temperature scaling rule by describing a simple yet insightful task which is not solvable by a single attention head unless temperature scaling is employed. Specifically, we consider a setting where the sequence exhibits *feature imbalances* where frequent tokens start dominating the context and potentially overwhelm the less frequent but relevant tokens.

Imbalanced token setup: Suppose the input sequence $\boldsymbol{X} = [\boldsymbol{x}_1 \ \dots \ \boldsymbol{x}_L]^\top$ is composed of a minority token $\boldsymbol{a} \in \mathbb{R}^d$ and a majority token $\boldsymbol{b} \in \mathbb{R}^d$, that is, $\boldsymbol{x}_i \in \{\boldsymbol{a}, \boldsymbol{b}\}$ for all $i \in [L]$. For each position, we will simply ask the model to output a target mixture of $\boldsymbol{a}$ and $\boldsymbol{b}$, namely, $\boldsymbol{y} = \alpha\boldsymbol{a} + (1-\alpha)\boldsymbol{b}$ for some $\alpha \in (0, 1)$. Thus, using a 1-layer causal attention, we study the following objective by calculating the loss between target $\boldsymbol{y}$ and each attention output:

$$\mathcal{L}(\boldsymbol{W}) = \frac{1}{L}\sum_{n=n_0}^{L} \|\boldsymbol{y} - \boldsymbol{X}^\top \mathbb{S}_{\le n}(\tau_n \cdot \boldsymbol{X}\boldsymbol{W}^\top \boldsymbol{x}_n)\|_2^2. \tag{1}$$

Above, τ_n is the inverse-temperature for the n^{th} position. Here, n_0 is a burn-in period to simplify our exposition: n_0 is the smallest number such that both $\boldsymbol{a}$ and $\boldsymbol{b}$ appear at least once within the first n_0 tokens[1]. Additionally, let n_a be the number of tokens $\boldsymbol{x}_i$ that are equal to $\boldsymbol{a}$ within $i \in [n]$. We have the following theorem.

Proposition 2. *Assume $\boldsymbol{a}, \boldsymbol{b}$ are unit Euclidean norm and linearly independent. Define the imbalance ratio $\kappa_n = (n - n_a)/n_a$ for $n \in [L]$. There is a $\boldsymbol{W}_\star$ such that, setting $\tau_n = \log \kappa_n + \log \frac{\alpha}{1-\alpha}$, $\mathcal{L}(\boldsymbol{W}_\star)$ minimizes the risk (1) to achieve $\mathcal{L}(\boldsymbol{W}_\star) = 0$.*

Conversely, consider the problem instance with target mixture *of $\alpha = 1/2$,* second-quadrant imbalance *of $2 \ge \kappa_n \ge 1$ for $L/4 \le n \le L/2$ and* fourth-quadrant imbalance *of $\kappa_n \ge 4$ for $n \ge 3L/4$. If we employ flat temperature $\tau_n = 1$ for all $n \in [L]$, for any choice of attention weights $\boldsymbol{W} \in \mathbb{R}^{d\times d}$, we have the lower bound $\mathcal{L}(\boldsymbol{W}) > 1/500$.*

[1]This is without loss of generality, otherwise, causal attention would output a fixed vector (either $\boldsymbol{a}$ or $\boldsymbol{b}$) regardless of the attention weights.

Table 2: We apply normalization to attention output and compute the MSE risk.

Vanilla	Value-selective	Naive averaging	Bayes optimal estimator
1.390	0.071	2.058	0.003

Proposition 2 inspired our design of position-aware temperature scaling. Intuitively, as n increases, the sequence may include less related tokens, leading to an increase in κ_n. When κ_n follows power-law $\kappa_n = n^{\texttt{pow}}$, we recover the logarithmic temperature scaling rule of $\tau_n = \texttt{const} + \texttt{pow} \cdot \log n$. Consequently, our Position-aware Temperature Scaling function τ_n is designed as $\tau^{pos}(\boldsymbol{x}) = 1 + \sigma(\alpha) log(n)$, n is the position length, α is the trainable parameter, σ is the non-linearity function sigmoid. The function is motivated by, other paper's rules [33, 26, 22, 52].

4.3 The benefits of incorporating value embedding

Within attention, value embeddings ($\boldsymbol{V}$) are transformed using only a linear projection. Consequently, each token's contribution to the output is a weighted sum based on the attention scores, with these weights adjusted linearly. In sequences with many tokens, irrelevant or noisy tokens can negatively influence the attention mechanism. Because value embeddings are linearly projected, they may not be able to fully distinguish between relevant and irrelevant tokens. The value-temperature scaling acts as a nonlinear scalar weighting function. By adjusting the temperature, we aim to control the impact of each token, suppressing the influence of irrelevant or noisy tokens. This helps emphasize more relevant tokens, thereby improving the quality of the context representation. We motivate the potential benefits of TS on value embeddings through the following synthetic denoising task.

Denoising task Let $[K]$ be the token alphabet with embeddings $(\boldsymbol{e}_i)_{i=1}^K$. Assume $d = K$ and $\boldsymbol{e}_i$'s are standard basis. Consider the following data distribution $(\boldsymbol{X}, \boldsymbol{y}) \sim \mathcal{D}$ where $\boldsymbol{X} = [\boldsymbol{x}_1 \ \dots \ \boldsymbol{x}_L]^\top \in \mathbb{R}^{L\times d}$ is the input sequence and $\boldsymbol{y} \in \mathbb{R}^d$ is the target label.

- Draw $q \sim \texttt{Unif}([K])$. Set $\boldsymbol{y} = \boldsymbol{e}_q$.
- Let $(\boldsymbol{z}_i)_{i=1}^L$ be IID noise vectors with $\mathcal{N}(0, \sigma^2 \boldsymbol{I})$
- $\boldsymbol{x}_L = \boldsymbol{e}_q + \boldsymbol{z}_L$. For $i \in [L-1]$, $\boldsymbol{x}_i$ is determined by a Bernoulli distribution with a parameter of α, selecting between $\boldsymbol{e}_q + \boldsymbol{z}_i$ and $\boldsymbol{z}_i$. Consequently, α of the tokens are signal tokens $\boldsymbol{e}_q + \boldsymbol{z}_i$.

The denoising objective is minimizing the MSE risk

$$\mathcal{L}(f) = \mathbb{E}_{\mathcal{D}}[\|\boldsymbol{y} - \text{norm}(\hat{\boldsymbol{y}})\|_2^2]$$

where $\text{norm}(\hat{\boldsymbol{y}}) = \hat{\boldsymbol{y}}/\|\hat{\boldsymbol{y}}\|_2$, $\hat{\boldsymbol{y}}$ is the output of model $f(\cdot)$, $\hat{\boldsymbol{y}} = f(\boldsymbol{X})$.

To solve this task, the attention model $f(\boldsymbol{X})$ should intelligently combine the tokens within $\boldsymbol{X}$ to approximate the denoised target $\boldsymbol{e}_q$. Importantly, the model will strictly benefit from eliminating the pure noise tokens, i.e., instances with $\boldsymbol{x}_i = \boldsymbol{z}_i$. Note that the value projection of the attention matrix will not suffice to denoise the input sequence. The reason is that q is uniform, and signal tokens span the whole space. Thus, we will benefit from a nonlinear denoising procedure.

To test this intuition, we use a 1-layer single-head attention model, denoted as different $f(\cdot)$ to minimize the denoising objective. We compare the model with value-selectivity to the following baselines:

1. *Vanilla Attention:* The standard 1-layer single-head attention model, $\hat{\boldsymbol{y}}_{att} = \text{Att}(\boldsymbol{X})$
2. *Value-selective self-attention:* 1-layer Selective Self-Attention (SSA). $\hat{\boldsymbol{y}}_{SSA} = \text{SSA}(\boldsymbol{X})$. Since this is a synthetic task, as a proxy for the token-aware temperature scaling, we use the selection function $\max_{j\in[d]} x_{ij} \geq 1/2$. Intuitively, when noise $\sigma \lesssim 1/\sqrt{\log d}$, thresholding with the largest entry will detect the signal tokens.
3. *Naive averaging:* Directly average the tokens, $\hat{\boldsymbol{y}}_{naive} = \frac{1}{L}\sum_{i=1}^L \boldsymbol{x}_i$.
4. *Bayes optimal estimator:* $\hat{\boldsymbol{y}}_{opt} = \frac{1}{|S|}\sum_{i\in S} \boldsymbol{x}_i$ where $S \subset [L]$ is the ground-truth set of signal tokens distributed as $\boldsymbol{e}_q + \boldsymbol{z}_i$.

The resulting MSE risks are displayed in Table 2. We set $d = k = 8$ and $\alpha = \frac{1}{4}$. With the addition of the value-selection function, the model achieved a loss comparable to the optimal estimator, indicating

Table 3: Experiment results for model pretraining and finetuning. For perplexity (ppl), lower is better, and for accuracy (acc), higher is better.

Model	Wikitext ppl↓	Lambada_std ppl↓	Lambada_openai ppl↓	Lambada_std acc↑	Lambada_openai acc↑	Piqa acc↑	Hella acc_norm↑	Winogrande acc↑	Arc-E acc↑	Arc-C acc_norm↑	Average acc↑
Finetune											
GPT2	36.503	51.631	29.134	0.340	0.451	0.584	0.313	0.476	0.457	0.221	0.406
+SSA base	34.618	50.412	27.235	0.361	0.469	0.610	0.338	0.512	0.479	0.249	0.431
+SSA weight sharing	35.147	50.832	27.905	0.357	0.465	0.603	0.334	0.500	0.472	0.243	0.425
Pythia-160m	26.681	47.996	24.102	0.383	0.494	0.674	0.362	0.542	0.503	0.277	0.462
+SSA base	26.514	47.945	23.956	0.388	0.513	0.688	0.375	0.557	0.530	0.291	0.477
+SSA weight sharing	26.780	47.961	24.027	0.386	0.509	0.685	0.369	0.553	0.524	0.285	0.473
Pythia-410m	20.310	42.694	21.895	0.418	0.542	0.696	0.372	0.547	0.561	0.288	0.489
+SSA base	19.976	42.689	21.704	0.430	0.553	0.714	0.381	0.558	0.572	0.302	0.501
+SSA weight sharing	20.190	42.692	21.810	0.428	0.549	0.707	0.380	0.551	0.566	0.295	0.497
Llama	19.764	28.023	16.513	0.426	0.574	0.704	0.377	0.549	0.595	0.302	0.504
+SSA base	19.305	27.627	15.860	0.428	0.581	0.710	0.388	0.562	0.618	0.336	0.518
+SSA weight sharing	19.512	27.892	16.038	0.426	0.579	0.708	0.385	0.557	0.608	0.331	0.513
Llama3-8b	12.416	24.002	13.954	0.481	0.684	0.772	0.544	0.698	0.780	0.463	0.632
+SSA base	10.982	23.671	12.052	0.489	0.690	0.779	0.550	0.703	0.787	0.472	0.639
+SSA weight sharing	11.498	23.805	10.164	0.487	0.687	0.776	0.548	0.701	0.784	0.471	0.636
Pretrain											
GPT2	35.813	104.225	42.187	0.216	0.304	0.608	0.309	0.462	0.359	0.186	0.349
+SSA base	33.528	103.933	40.960	0.221	0.318	0.631	0.317	0.480	0.365	0.203	0.362
+SSA weight sharing	34.601	104.004	41.326	0.219	0.312	0.622	0.312	0.469	0.365	0.197	0.356
Pythia-160m	27.943	75.487	34.406	0.279	0.351	0.630	0.348	0.498	0.401	0.219	0.389
+SSA base	26.912	72.891	33.126	0.294	0.360	0.661	0.359	0.508	0.426	0.230	0.405
+SSA weight sharing	27.046	73.071	33.814	0.291	0.360	0.660	0.352	0.503	0.421	0.221	0.401
Pythia-410m	22.516	69.814	32.781	0.321	0.371	0.655	0.357	0.530	0.441	0.234	0.416
+SSA base	21.402	68.553	31.269	0.336	0.387	0.660	0.363	0.536	0.449	0.237	0.424
+SSA weight sharing	21.980	69.041	31.458	0.331	0.384	0.658	0.362	0.534	0.445	0.237	0.422

successful suppression of noisy tokens. In contrast, while vanilla softmax self-attention performs similarly to naive averaging, it fails to sufficiently denoise, resulting in a much larger loss compared to our value-selective attention.

5 Empirical Evaluations

5.1 Standard Benchmarks

Drawing on theoretical insights, we assess the performance of SSA on NLP tasks by integrating SSA into established models such as GPT-2 [34], Pythia [3], Llama [44] and Llama3 [16]. Our methodology includes both pre-training and fine-tuning to evaluate SSA's performance and efficiency. For the pre-training evaluation, we train the model from scratch on the SlimPajama dataset [41]. Subsequently, we evaluate the model on various downstream zero-shot tasks, including Wikitext [27], Lambada [32], Piqa [4], Hella [51], Winogrande [38], Arc-E, and Arc-C [10]. This approach is widely used for measuring the performance and generalization capabilities of pretrained large language models across diverse tasks [2, 3, 18]. For the fine-tuning evaluation, we start by loading the official pre-trained model and then fine-tune it on the downstream tasks. Unlike pre-training, where the downstream tasks are unseen during training, fine-tuning involves direct training on the tasks. This allows the model to better approximate the token distribution and understand the text domain. Details of the models are provided in Appendix A.

Our primary results are shown in Table 3. Based on the theoretical insights and ablation study results, we conduct both Token-aware and Position-aware Temperature Scaling on query $\boldsymbol{Q}$, and value $\boldsymbol{V}$. We observe that across various models and datasets, incorporating SSA consistently enhances performance. Notably, experiments with larger and more recent models, such as Llama3-8B and Pythia 410M, confirm that SSA improves accuracy across across model scales and architectures. We further introduce a weight sharing strategy that reduces the number of parameter overhead to less than 0.5% while preserving the benefits of SSA and still outperforming the standard transformer. This underscores the value of selectivity irrespective of its precise implementation. Thus, our improvements are not arising from an increase in the parameter count, but rather from the strategic integration of SSA. Additionally, we have also explored a *feature-based* method to further enhance SSA's parameter efficiency. In a nutshell, rather than training an MLP, we select the temperature as a function of *token-level features*, such as the frequency of a token in the training corpus, by fitting a single scalar parameter. This process requires only O(1) additional weights (<0.01% of total). Further details and results are provided in Appendix B.2.

For the ablation study, we fine-tuned the models on the Wikitext dataset to compare the influence of each component, using the same dataset and training configurations as those in the real experiments. The results are shown in Appendix B.1. Among the results, we observe that deploying both Token-aware and Position-aware Temperature Scaling on $\boldsymbol{Q}$ and $\boldsymbol{V}$ independently could achieve significant improvement, aligning with our theoretical insights. Additionally, combining Key and Query temperatures can achieve additional improvement. Moreover, between token-aware and position-aware temperature scaling, the latter demonstrates a more consistent improvement across different scenarios, while combining them can achieve the best overall result. We also compare with more baselines including [26, 22, 52] and the results are shown in Appendix B.3. Our method consistently outperforms the baselines.

Additionally, SSA can accelerate the training process by achieving comparable performance with fewer tokens. This efficiency not only reduces the demand on computational resources but also shortens the time required to effectively train models. We illustrate this efficiency by plotting the training results when fine-tuning the Llama model on the Wikitext dataset, both with vanilla attention layer or SSA, in Figure 4. The results indicate that SSA can accelerate training, achieving similar performance with 1.45× reduction in pretraining steps.

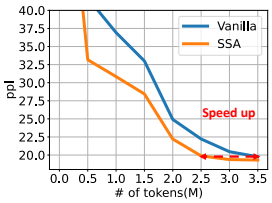


Figure 4: Comparison of training curves. SSA provides reasonable benefits in terms of training speedup.

5.2 Passkey Retrieval

We also examines the perfromance on the passkey retrieval task as defined in [33, 29].This is a synthetic task to measure a model's ability to retrieve a simple passkey (i.e., a five-digit number) within a large amount of otherwise meaningless text. We performed 10 iterations of the passkey retrieval task with the passkey placed at a random location uniformly distributed across the evaluation context window. Intuitively, SSA could better solve this task by assigning different token-level temperatures to digits vs words. For our evaluation of the fine-tuned Pythia, SSA leads to substantial improvement (from 56.9% to 74.4%), as seen in Table 4.

Table 4: Passkey retrieval performance of various models.

Model	Original	+SSA	+SSA(weight sharing)
Pythia-160m	56.89	74.41	66.90
Llama	77.62	89.53	89.45

6 Conclusions, Limitations, and Future Directions

We have introduced the Selective Self-Attention layer, which augments the softmax nonlinearity with a principled temperature-scaling strategy. SSA shows consistent benefits and augments the performance of existing transformer-based models such as Pythia and Llama 2. We also provide theoretical insights into the benefits of query, value, and positional selectivity.

Future research. Based on SSA, there are several interesting research avenues to pursue. Firstly, our method can extend to linear attention strategies. While we can use the same method for value embeddings, for queries, we can train an additive bias term on attention similarities rather than using temperature scaling. Secondly, based on the visual benefits of SSA on Figure 3, it would be interesting to explore how SSA can help the interpretability and quality of the attention maps. Overall, SSA has the potential to assist in more principled use of transformers in language, vision, and other modalities.

Limitations. Our work focuses on the canonical softmax-attention mechanism, which suffers from the quadratic computation bottleneck. As mentioned above, extending our method to linear attention can mitigate computational costs. Another direction to enhance efficiency is building stronger connections to sparsity and understanding how SSA can benefit and be integrated with sparse attention algorithms.

Acknowledgements

This work was supported in part by the National Science Foundation grants CCF-2046816, CCF-2403075, CCF-2008020, the Office of Naval Research award N000142412289, an Adobe Data Science Research award, and gifts by Open Philanthropy and Google Research.

References

[1] Emmanuel Abbe, Samy Bengio, Aryo Lotfi, and Kevin Rizk. Generalization on the unseen, logic reasoning and degree curriculum. In *International Conference on Machine Learning*, pages 31–60. PMLR, 2023.

[2] Simran Arora, Sabri Eyuboglu, Michael Zhang, Aman Timalsina, Silas Alberti, Dylan Zinsley, James Zou, Atri Rudra, and Christopher Ré. Simple linear attention language models balance the recall-throughput tradeoff. *arXiv preprint arXiv:2402.18668*, 2024.

[3] Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR, 2023.

[4] Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439, 2020.

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

[6] Beidi Chen, Tri Dao, Eric Winsor, Zhao Song, Atri Rudra, and Christopher Ré. Scatterbrain: Unifying sparse and low-rank attention. *Advances in Neural Information Processing Systems*, 34:17413–17426, 2021.

[7] Ta-Chung Chi, Ting-Han Fan, and Alexander I Rudnicky. Attention alignment and flexible positional embeddings improve transformer length extrapolation. *arXiv preprint arXiv:2311.00684*, 2023.

[8] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

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

[10] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

[11] Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*, 2023.

[12] Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359, 2022.

[13] Yann N Dauphin, Angela Fan, Michael Auli, and David Grangier. Language modeling with gated convolutional networks. In *International conference on machine learning*, pages 933–941. PMLR, 2017.

[14] Soham De, Samuel L Smith, Anushan Fernando, Aleksandar Botev, George Cristian-Muraru, Albert Gu, Ruba Haroun, Leonard Berrada, Yutian Chen, Srivatsan Srinivasan, et al. Griffin: Mixing gated linear recurrences with local attention for efficient language models. *arXiv preprint arXiv:2402.19427*, 2024.

[15] Yihe Dong, Jean-Baptiste Cordonnier, and Andreas Loukas. Attention is not all you need: Pure attention loses rank doubly exponentially with depth. In *International Conference on Machine Learning*, pages 2793–2803. PMLR, 2021.

[16] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[17] Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

[18] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

[19] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[20] M Emrullah Ildiz, Yixiao Huang, Yingcong Li, Ankit Singh Rawat, and Samet Oymak. From self-attention to markov models: Unveiling the dynamics of generative transformers. *International Conference on Machine Learning*, 2024.

[21] Tobias Katsch. Gateloop: Fully data-controlled linear recurrence for sequence modeling. *arXiv preprint arXiv:2311.01927*, 2023.

[22] Mingchen Li, Xuechen Zhang, Christos Thrampoulidis, Jiasi Chen, and Samet Oymak. Autobalance: Optimized loss functions for imbalanced data. *Advances in Neural Information Processing Systems*, 34:3163–3177, 2021.

[23] Bingbin Liu, Jordan Ash, Surbhi Goel, Akshay Krishnamurthy, and Cyril Zhang. Exposing attention glitches with flip-flop language modeling. *Advances in Neural Information Processing Systems*, 36, 2024.

[24] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[25] Harsh Mehta, Ankit Gupta, Ashok Cutkosky, and Behnam Neyshabur. Long range language modeling via gated state spaces. In *International Conference on Learning Representations*, 2023.

[26] Aditya Krishna Menon, Sadeep Jayasumana, Ankit Singh Rawat, Himanshu Jain, Andreas Veit, and Sanjiv Kumar. Long-tail learning via logit adjustment. *ICLR*, 2021.

[27] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

[28] Tomas Mikolov. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[29] Amirkeivan Mohtashami and Martin Jaggi. Landmark attention: Random-access infinite context length for transformers. *arXiv preprint arXiv:2305.16300*, 2023.

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

[31] Matteo Pagliardini, Daniele Paliotta, Martin Jaggi, and François Fleuret. Fast attention over long sequences with dynamic sparse flash attention. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[32] Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The lambada dataset: Word prediction requiring a broad discourse context. *arXiv preprint arXiv:1606.06031*, 2016.

[33] Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. Yarn: Efficient context window extension of large language models. *arXiv preprint arXiv:2309.00071*, 2023.

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

[35] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

[36] Liliang Ren, Yang Liu, Shuohang Wang, Yichong Xu, Chenguang Zhu, and ChengXiang Zhai. Sparse modular activation for efficient sequence modeling. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[37] Arda Sahiner, Tolga Ergen, Batu Ozturkler, John Pauly, Morteza Mardani, and Mert Pilanci. Unraveling attention via convex duality: Analysis and interpretations of vision transformers. In *International Conference on Machine Learning*, pages 19050–19088. PMLR, 2022.

[38] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

[39] Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*, 2021.

[40] Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

[41] Zhiqiang Shen, Tianhua Tao, Liqun Ma, Willie Neiswanger, Joel Hestness, Natalia Vassilieva, Daria Soboleva, and Eric Xing. Slimpajama-dc: Understanding data combinations for llm training. *arXiv preprint arXiv:2309.10818*, 2023.

[42] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[43] Davoud Ataee Tarzanagh, Yingcong Li, Xuechen Zhang, and Samet Oymak. Max-margin token selection in attention mechanism. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

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

[45] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[46] Junxiong Wang, Tushaar Gangavarapu, Jing Nathan Yan, and Alexander M Rush. Mambabyte: Token-free selective state space model. *arXiv preprint arXiv:2401.13660*, 2024.

[47] Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit bayesian inference. *arXiv preprint arXiv:2111.02080*, 2021.

[48] Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated linear attention transformers with hardware-efficient training. *arXiv preprint arXiv:2312.06635*, 2023.

[49] Shunyu Yao, Binghui Peng, Christos Papadimitriou, and Karthik Narasimhan. Self-attention networks can process bounded hierarchical languages. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3770–3785, Online, August 2021. Association for Computational Linguistics.

[50] Tianzhu Ye, Li Dong, Yuqing Xia, Yutao Sun, Yi Zhu, Gao Huang, and Furu Wei. Differential transformer. *arXiv preprint arXiv:2410.05258*, 2024.

[51] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

[52] Xuechen Zhang, Mingchen Li, Jiasi Chen, Christos Thrampoulidis, and Samet Oymak. Class-attribute priors: Adapting optimization to heterogeneity and fairness objective. *to appear at AAAI*, 2024.

[53] Hattie Zhou, Arwen Bradley, Etai Littwin, Noam Razin, Omid Saremi, Josh Susskind, Samy Bengio, and Preetum Nakkiran. What algorithms can transformers learn? a study in length generalization. *arXiv preprint arXiv:2310.16028*, 2023.

[54] Lianghui Zhu, Bencheng Liao, Qian Zhang, Xinlong Wang, Wenyu Liu, and Xinggang Wang. Vision mamba: Efficient visual representation learning with bidirectional state space model. *arXiv preprint arXiv:2401.09417*, 2024.

Table 5: Fine-tuning experiment results for language models on the Wikitext dataset, showcasing baseline and variations with different components ($\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}$).

Configuration	Pythia	GPT2
Baseline	28.781	36.503
$\boldsymbol{Q}$	27.416	34.832
$\boldsymbol{K}$	28.715	36.443
$\boldsymbol{V}$	27.980	35.857
$\boldsymbol{Q}, \boldsymbol{V}$	26.514	34.618
$\boldsymbol{K}, \boldsymbol{Q}, \boldsymbol{V}$	26.603	34.609

A Implementation details

For the next token prediction experiment that substantiates the expressivity benefit of query-selectivity, as detailed in Figure 3 and Table 1, we employ the Adam optimizer to train a model. This model consists of a single-layer, single-head attention mechanism, accompanied by a tokenizer and a fully connected layer. The tokenizer embeds the discrete sequence to continuous embedding $\boldsymbol{E}$. The fully connected layer is used as the classifier to predict the node index. We set the learning rate at $1e^{-4}$. The training loss is the cross-entropy loss. In our experiments, $L = N = 8$. For SSA, we implement Token-aware Temperature Scaling for the query matrix $\boldsymbol{Q}$. We assign a scaling parameter to each group of nodes that share the same number of neighbors. To have better visualization, we do normalization to plot the attention map $\boldsymbol{P}_\star$ and $\hat{\boldsymbol{P}}$. For the experiments shown in Table 2, we also use the Adam optimizer and learning rate $1e^{-4}$. But the objective is the MSE risk.

For our empirical evaluation, we utilize several models. We employ GPT-2, which has 124 million parameters, and use the official OpenAI GPT-2 checkpoints that were pre-trained on the WebText dataset [34] for our finetuning experiments. For Pythia, our experiments are conducted with a model size of 160 million and 410 million parameters, using the official checkpoint pre-trained on the Pile dataset [17]. Lastly, for Llama, we utilize the smallest variant available, with 7 billion parameters, and similarly fine-tune using the official pre-trained model. As the training configuration, we train with 3.5 million tokens for fine-tuning and 15B tokens for pre-training. We always use the AdamW optimizer [24], $\beta_1 = 0.9$ and $\beta_2 = 0.95$. We set learning rate $1e^{-6}$ with no weight decay and no warmup. The pre-training takes about 2 hours using 4 A40 and fine-tuning takes about 2 days. We use FlashAttention [11] to accelerate the training. For weight sharing, each head shares the same funtion. All the experiments are conducted with 4 or 8 A40 or L40S. We can directly reuse FlashAttention[12], which significantly improves model efficiency.

B Additional experiments

B.1 Ablation study

For the ablation study, we conducted fine-tuning on the Wikitext dataset to compare the influence of each component, using the same dataset and training configurations as those in the real experiments. The models are evaluated with perplexity (ppl).

B.1.1 Key-temperature, Query-temperature, and Value-temperature

To evaluate the benefits of applying temperature scaling to $\boldsymbol{K}$, $\boldsymbol{Q}$, and $\boldsymbol{V}$, we conducted an ablation study, examining each component individually and in combination. For a fair comparison, both position-aware and token-aware temperature scaling were applied to all components. The results, detailed in table 5, indicate that modifying $\boldsymbol{Q}$, and $\boldsymbol{V}$ independently yields clear benefits, whereas alterations to $\boldsymbol{K}$ result in performance that is similar to, or even worse than, the baseline vanilla attention layer. The results align well with the theoretical analysis presented in section 4. However, when $\boldsymbol{Q}$ and $\boldsymbol{V}$ are combined, we observe consistent improvements. These findings led us to develop our final algorithm, which applies temperature scaling to both $\boldsymbol{Q}$ and $\boldsymbol{V}$.

Table 6: Investigate the benefits of Token-aware Temperature Scaling, Position-aware Temperature Scaling.

model	vanilla	$\boldsymbol{q}$			$\boldsymbol{v}$		
		$\tau^{pos} + \tau^{tok}$	τ^{pos}	τ^{tok}	$\tau^{pos} + \tau^{tok}$	τ^{pos}	τ^{tok}
Pythia	28.781	27.416	27.995	27.503	27.980	28.342	27.975
GPT2	36.503	34.832	34.970	35.064	35.857	36.320	35.617

Table 7: Comparing different SSA parameterizations

Model	Wikitext ppl↓	Lambada_std ppl↓	Lambada_openai ppl↓	Lambada_std acc↑	Lambada_openai acc↑
Finetune					
Pythia	26.681	47.996	24.102	0.383	0.494
Pythia +SSA base	26.514	47.945	23.956	0.388	0.513
Pythia +SSA weight sharing	26.780	47.961	24.027	0.386	0.509
Pythia + SSA feature-based	27.048	47.966	24.114	0.387	0.499
Pretrain					
Pythia	27.943	75.487	34.406	0.279	0.351
Pythia +SSA base	26.912	72.891	33.126	0.294	0.360
Pythia +SSA weight sharing	27.046	73.071	33.814	0.291	0.360
Pythia +SSA feature-based	27.281	73.614	33.794	0.287	0.357

B.1.2 Token-aware Temperature Scaling, Position-aware Temperature Scaling

We also conduct experiments to investigate the benefits of Token-aware and position-aware Temperature Scaling applied to Query $\boldsymbol{q}$ and value $\boldsymbol{v}$. The results are shown in table 6. Token-aware Temperature Scaling positively impacts both Query $\boldsymbol{q}$ and value $\boldsymbol{v}$, whereas Position-aware Temperature Scaling shows smaller improvement on value $\boldsymbol{v}$, aligning with our theoretical insights. Furthermore, when compared to GPT-2, Pythia—which features a more advanced positional encoding[42] —demonstrates fewer improvements. This suggests that while new strategies may mitigate the dispersed attention issue, our Selective Self-Attention (SSA) method still offers additional improvements.

B.2 Parameter-efficient SSA: Weight sharing and featurization

Here, we also introduce a feature-based approach to improve parameter efficiency. In a nutshell, rather than training an MLP, we select the temperature as a function of *token-level features*, such as the frequency of a token in the training corpus, by fitting a single scalar parameter. This process requires only O(1) additional weights per attention head (<0.01% of total). This is inspired by the logit adjustment strategy of [26] which sets the cross-entropy temperature as a function of class frequencies.

Our evaluations on feature-based SSA are provided in Table 7. We find that, while the feature-based method is beneficial and highly parameter-efficient, it can be sensitive to feature selection and exhibits more variability across datasets.

B.3 Ablation of Different Parameterizations

In addition to the functions we propose for temperature design, we also explore alternative approaches. Instead of employing our Position-aware Temperature Scaling function, we use a constant parameter, as suggested in other studies [26, 22, 52]. We also compare with the temperature scaling method proposed by [33]. Furthermore, in place of our Token-aware Temperature Scaling, we adopt a simpler approach by directly utilizing token frequency and training only a scale parameter. These experiments were conducted using the Pythia model, fine-tuned on the Wikitext dataset. The outcomes of these comparative analyses are presented in Table 8. Among those baselines, we consistently outperform their results.

Table 8: Conducting different functions.

Configuration		Pythia
Vanilla		28.781
Token	Yarn [33]	27.602
	Constant	28.058
Position	Frequency	27.360
Position+Token	SSA	26.514

C Proofs

C.1 Proof of Proposition 1

Proof. Given embeddings $\boldsymbol{e}_1, \boldsymbol{e}_2$ with unit ℓ_2 norm, their correlation is $\rho = \boldsymbol{e}_1^\top \boldsymbol{e}_2$.

First, consider the approximation error bound $\|\boldsymbol{P}_\star - \mathbb{S}(\boldsymbol{E}\boldsymbol{W}\boldsymbol{E}^\top)\|_\infty \le \varepsilon$. To achieve this, the weight matrix $\boldsymbol{W}$ must satisfy the inequality.

To derive a lower bound on $\|\boldsymbol{W}\|$, observe that:

$$\begin{aligned} \|\boldsymbol{P}_\star - \mathbb{S}(\boldsymbol{E}\boldsymbol{W}\boldsymbol{E}^\top)\|_\infty &\le \varepsilon \\ \|\boldsymbol{P}_\star - \mathbb{S}(\boldsymbol{E}\boldsymbol{W}\boldsymbol{E}^\top)\|_\infty &= \left\| \frac{1}{1+e^{-\boldsymbol{E}\boldsymbol{W}\boldsymbol{E}^\top}} - \boldsymbol{P}_\star \right\|_\infty \\ &\le \varepsilon \end{aligned}$$

Using the fact that $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ are unit vectors and $\rho = \boldsymbol{e}_1^\top \boldsymbol{e}_2$, we have:

$$\|\boldsymbol{E}\boldsymbol{W}\boldsymbol{E}^\top\|_\infty \ge \frac{1}{4\varepsilon} - \Gamma.$$

Now, the norm $\|\boldsymbol{W}\|$ is given by:

$$\|\boldsymbol{W}\| \ge \frac{\|\boldsymbol{e}_1 - \boldsymbol{e}_2\|^{-1}}{\sqrt{2-2\rho^2}} \left(\log\left(\frac{1}{4\varepsilon}\right) - \Gamma \right).$$

Conversely, to achieve ε-approximation using Selective Attention, the weights need to be bounded such that:

$$\tau(\boldsymbol{e}_{1,2}) \cdot \|\boldsymbol{W}\| \le \|\boldsymbol{e}_1 - \boldsymbol{e}_2\|^{-1} \max\left(\log\left(\frac{1}{\varepsilon}\right), \frac{\Gamma}{\sqrt{1-\rho^2}} \right).$$

Therefore, the selective attention avoids the $1/\sqrt{1-\rho^2}$ dependence on the $\log(1/\varepsilon)$ term, decoupling the high-specificity requirement (small ε) from the semantic similarity of the tokens.

This completes the proof of Proposition 1. □

C.2 Proof of Proposition 2

Proof. We first show the success direction. Set $\boldsymbol{W}$ such that $\boldsymbol{b}^\top \boldsymbol{W} = 0$ and $\boldsymbol{a}^\top \boldsymbol{W} \boldsymbol{a} = \boldsymbol{a}^\top \boldsymbol{W} \boldsymbol{b} = 1$. Such $\boldsymbol{W}$ exists thanks to the linear independence of $\boldsymbol{a}, \boldsymbol{b}$. Now plugging this $\boldsymbol{W}$ into (1), for each position, regardless of whether $\boldsymbol{x}_n = \boldsymbol{a}$ or $\boldsymbol{x}_n = \boldsymbol{b}$, we obtain

$$\boldsymbol{X}^\top \mathbb{S}_{\le n}(\tau_n \cdot \boldsymbol{X}\boldsymbol{W}\boldsymbol{x}_n) = \frac{n_a e^{\tau_n}}{n_a e^{\tau_n} + (n - n_a)} \boldsymbol{a} + \frac{(n-n_a)e^{\tau_n}}{n_a e^{\tau_n} + (n-n_a)} \boldsymbol{b}.$$

We wish to ensure

$$\frac{n_a e^{\tau_n}}{n_a e^{\tau_n} + (n-n_a)} = \frac{1}{1 + (n/n_a - 1)e^{-\tau_n}} = \frac{1}{1+\kappa_n e^{-\tau_n}} = \alpha.$$

This in turn implies

$$\kappa_n e^{-\tau_n} = \frac{1-\alpha}{\alpha} \iff \tau_n = \log \kappa_n + \log \frac{\alpha}{1-\alpha}.$$

Next, we discuss the failure case of flat temperature. To do so, we will lower bound the loss over the queries $\boldsymbol{x}_n = \boldsymbol{b}$. Set $M = e^{(\boldsymbol{b}-\boldsymbol{a})^\top \boldsymbol{W}\boldsymbol{b}}$. Following same argument as above, for fixed temperature, $\boldsymbol{W}$ will output a non-adaptive composition of the form

$$\boldsymbol{X}^\top \mathbb{S}_{\leq n}(\boldsymbol{X}\boldsymbol{W}\boldsymbol{b}) = \frac{1}{1+M\kappa_n}\boldsymbol{a} + \frac{M\kappa_n}{1+M\kappa_n}\boldsymbol{b}.$$

Thus, the loss function will be lower bounded by (accounting for the prediction error in $\boldsymbol{a}, \boldsymbol{b}$ terms and their orthogonality)

$$\mathcal{L}(\boldsymbol{W}) \geq \min_{M>0} \sum_{n:\boldsymbol{x}_n=\boldsymbol{b}} (0.5 - \frac{1}{1+M\kappa_n})^2 + (0.5 - \frac{M\kappa_n}{1+M\kappa_n})^2 = \min_{M>0} \sum_{n:\boldsymbol{x}_n=\boldsymbol{b}} 2\cdot(0.5 - \frac{1}{1+M\kappa_n})^2.$$

Now since $\kappa_n \geq 1$ over both second, at least $1/2$ of the queries are $\boldsymbol{x}_n = \boldsymbol{b}$. Similarly, at least $4/5$ of the queries are $\boldsymbol{x}_n = \boldsymbol{b}$ over the last quadrant. We will lower bound the loss over the two scenarios depending on $M \geq 1/3$ or not.

First, suppose $M \geq 1/3$, in that case, using $\kappa_n \geq 4$ over the last quadrant, the loss is lower bounded by

$$\mathcal{L}(\boldsymbol{W}) \geq \min_{M>0} \frac{1}{N} \sum_{n\geq 3N/4, \boldsymbol{x}_n=\boldsymbol{b}} 2\cdot(0.5 - \frac{1}{1+M\kappa_n})^2 \geq \frac{2}{5}(0.5 - \frac{1}{1+4/3})^2 > 0.002.$$

where we used the fact that there are $\geq \frac{N}{5} = \frac{N}{4}\cdot\frac{4}{5}$ queries with $\boldsymbol{x}_n = \boldsymbol{b}$ over the last quadrant.

Similarly, suppose $M \leq 1/3$, in that case, using $\kappa_n \leq 2$ over second quadrant, the loss is lower bounded by

$$\mathcal{L}(\boldsymbol{W}) \geq \min_{M>0} \frac{1}{N} \sum_{N/2\geq n\geq N/4, \boldsymbol{x}_n=\boldsymbol{b}} 2\cdot(0.5 - \frac{1}{1+M\kappa_n})^2 \geq \frac{1}{4}(0.5 - \frac{1}{1+2/3})^2 = 0.0025.$$

where we used the fact that there are at least $N/8$ queries with $\boldsymbol{x}_n = \boldsymbol{b}$ over the second quadrant. Combining these two cases, we found that, for any choice of $\boldsymbol{W}$, the loss is lower bounded by 0.002. □

D Further Discussion on the Sparsity and Temperature Connection

Connections between sparsity and temperature.

To formally study the sparsity and temperature connection, let us consider a fixed attention row n and introduce:

- $\boldsymbol{s}(\tau) = \mathbb{S}_{\leq n}(\tau \cdot \boldsymbol{X}\boldsymbol{W}\boldsymbol{x}_n)$, the scaled attention scores with inverse-temperature $\tau > 0$.
- $\bar{\boldsymbol{s}}(\kappa) = \mathbb{S}_{\leq n}(\boldsymbol{X}\boldsymbol{W}\boldsymbol{x}_n, \kappa)$ denote the sparse attention scores where the top-κn entries are retained and the rest are set to 0 where $0 \leq \kappa \leq 1$.

The connection between sparsity and temperature scaling is clear. For instance, the top entry of $\boldsymbol{s}(\tau)$ will be decreasing in τ whereas the entropy of $\boldsymbol{s}(\tau)$ will be increasing. Here, we would like to establish how temperature scaling rule can be mapped to a sparsity rule. We will do this under a power-law relevance assumption on the attention scores. Here, we assume that the attention scores admit two values and the fraction of larger/relevant attention scores follow a power-law as context window grows.

Assumption 1 (Power-law relevance)**.** *Consider the vector of raw attention scores $\boldsymbol{a} = \boldsymbol{X}\boldsymbol{W}\boldsymbol{x}_n$. Each entry of $\boldsymbol{a}$ is either c or $c_+ = c + \gamma$ for some $\gamma > 0$. Additionally, $n^{-\texttt{pow}}$ fraction of the entries are equal to c_+ for some $\texttt{pow} > 0$.*

Above c_+ is the score attained by the salient tokens, γ is the score advantage of salient tokens over rest of the tokens, and pow dictates the fraction of the salient tokens. To proceed, we have the following lemma which identifies condition under which TS and sparse-attention exhibit the same softmax temperature behavior.

Lemma 2. *For any choice of $\tau = 1 + \alpha \log n > 0$ and corresponding sparsity $\kappa = \frac{n^{-\alpha\gamma}}{1-n^{-\texttt{pow}}} + n^{-\texttt{pow}}$, we have that $\|\boldsymbol{s}(\tau)\|_{\ell_\infty} = \|\bar{\boldsymbol{s}}(\kappa)\|_{\ell_\infty}$.*

This reveals the clear connection between temperature scaling and sparsification rules. Simplifying the above, this lemma advocates that the sparsification rule should follow the power law decay of $\kappa \approx n^{-\alpha\gamma \wedge \mathtt{pow}}$. Consistent with this lemma, our experiments demonstrate that sparsification with power-law results in respectable performance.

Proof. We first compute the top entry of $\boldsymbol{s}(\tau)$ as follows

$$\frac{s_1^\tau}{\sum_{i=1}^n s_i^\tau} = \frac{e^{\gamma\tau}}{n^{1-\mathtt{pow}}e^{\gamma\tau} + (n - n^{1-\mathtt{pow}})} = \frac{1}{n^{1-\mathtt{pow}} + n(1 - n^{-\mathtt{pow}})e^{-\gamma\tau}}$$

We similarly compute the top sparse attention score as

$$\frac{s_1}{\sum_{i=1}^\kappa s_i} = \frac{e^{\gamma}}{n^{1-\mathtt{pow}}e^{\gamma} + (\kappa - n^{-\mathtt{pow}})n} = \frac{1}{n^{1-\mathtt{pow}} + (\kappa - n^{-\mathtt{pow}})e^{-\gamma}n}.$$

Combining these, the top softmax probabilities are matched by setting

$$(\kappa - n^{-\mathtt{pow}})e^{-\gamma} = (1 - n^{-\mathtt{pow}})e^{-\gamma\tau} \iff \kappa = \frac{e^{-\gamma(\tau-1)}}{1 - n^{-\mathtt{pow}}} + n^{-\mathtt{pow}}.$$

□

We need to clarify that our method is not about sparse approximation of the attention map and instead aims to control the "spikiness of attention". The "spikiness of attention" can be viewed as an "effective sparsity" which can be quantified through L_∞ norm, L_1/L_2 ratio, or inverse-entropy of the softmax map. This discussion will also better clarify what is meant by "contextual sparsity" throughout the paper and distinguish it from (hard) sparsity targeted in [31, 36].

E Theoretical Considerations

E.1 Hierarchical vocabulary

Hierarchical vocabulary. Consider a k-ary tree of depth D: Each node has exactly k children, except at depth d. Such a tree has $1 + k + k^2 + \cdots + k^D = N = (k^{D+1} - 1)/(k - 1)$ nodes. The tree will correspond to the words/tokens in the vocabulary $\mathcal{V}$ of size N.

Token generation rule: Let $X \in \mathcal{V}^L$ be a sequence of length L drawn from $\mathcal{V}$. Suppose X ends with $q := x_L$. The token $Y = x_{L+1}$ that follows X will be drawn from q or the children of q available in the context window. If q is at depth l, it can attend to a total of $(k^{D+1-l} - 1)/(k - 1)$ unique tokens, including itself. Let $\mathcal{D}_{XY}$ denote the data distribution (Y, X) where Y is drawn uniformly from one of the *child tokens of* x_L available in the context window X.

The claims below aim to formalize the benefits of SSA for modeling the hierarchical token generation process. Let $\boldsymbol{E} = [\boldsymbol{e}_1 \ \dots \ \boldsymbol{e}_N]^\top$ be the token embeddings associated to the vocabulary $\mathcal{V}$. Assume elements of $\boldsymbol{E}$ are unit ℓ_2 norm. During training, we embed the discrete sequence X into $\boldsymbol{X} = [\boldsymbol{e}_{x_1} \ \dots \ \boldsymbol{e}_{x_N}]$.

Claim 1 (Benefits on attention map)**.** *Consider the attention map* $\mathtt{map}(\boldsymbol{X}) = \mathbb{S}(\tau(\boldsymbol{x}_L) \cdot \boldsymbol{X}\boldsymbol{W}\boldsymbol{x}_L)$*. Note that* map *is a function of* $\boldsymbol{W}, \boldsymbol{E}, \tau$*. Define the ideal attention map* X *to be* $\mathtt{map}^\star(X)$ *which uniformly attends to the children of* x_L *and assigns zero probability to other tokens. Define the population error*

$$\mathtt{err_map}(\boldsymbol{E}, \boldsymbol{W}, \tau) = \mathbb{E}_{X \sim \mathcal{D}_X}\left[\|\mathtt{map}(\boldsymbol{X}) - \mathtt{map}^\star(X)\|_1\right].$$

Under suitable assumptions (see remark below), QSSA is provably better than vanilla self-attention i.e. having $\tau(\boldsymbol{x})$ *improves attention capability by reducing* $\mathtt{err_map}(\boldsymbol{E}, \boldsymbol{W}, \tau)$*.*

Claim 2 (Benefits on prediction)**.** *Let* $f(\boldsymbol{X})$ *be an attention layer (SSA or vanilla). Suppose we sample the next token* $\hat{Y}$ *from* $f(\boldsymbol{X})$ *according to the distribution* $g(\boldsymbol{X}) = \mathbb{S}(\boldsymbol{C}f(\boldsymbol{X})) \in \mathbb{R}^N$*. Here* $\boldsymbol{C} \in \mathbb{R}^{N \times d}$ *is the linear prediction head. As loss measure, use the expected total-variation (TV) distance between* $g(\boldsymbol{X})$ *and the true label* Y*. Under suitable assumptions,* $g(\boldsymbol{X})$ *with QSSA* $f(\boldsymbol{X})$ *fits better to the hierarchical distribution compared to vanilla* $f(\boldsymbol{X})$*.*

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We introduce the studied question and list our contribution in Section 1.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitation of our work in Section 6

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: We provide our theory and related theoretical definitions in Section 4. We provide the related lemmas and proofs in Appendix.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide the design of our algorithm in Section 3 and implementation details in Section 5. We also provide main experimental results and detailed analysis in Section 5 and additional results in Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We release our training and evaluation code in a zip file.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We explain the training and testing details in Section 5

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We run our experiments on 2 representative datasets and compare our models performance with several baselines. The experiments results are 3 runs average and we provide error bar in the table.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: In Section 5, we state the hardware to run the experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: There is no potential harms caused by the research process and potential harmful societal impact.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We state the potential social benefit of our work in Section 6.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: We state the safeguards of our work in Section 6.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cite the dataset and also all the models used in our paper.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We include our experiment code and data generation code in our supplementary files.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: Our method and experiments only relate to next token generation, which does not involve human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: Our method and experiments only relate to next token generation, which does not involve human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.