Gaussian Process Attention with Kernel Modeling

Anonymous ACL submission

Abstract

Transformers dominate NLP, yet their core component, self-attention, remains a heuristic, lacking a robust theoretical foundation. This paper reinterprets self-attention with rotary positional embeddings (RoPE) as Nadaraya-Watson kernel regression, unlocking a novel framework for enhancing attention through kernel modeling. We introduce Gaussian Process Attention (GPA), which augments RoPE with a bank of decaying periodic kernels to capture linguistic patterns like periodicity and decay. Tested on a GPT model with character-level tokenization and a 13-million-character corpus, GPA outperforms baseline RoPE, reducing mean cross-entropy loss. GPA kernel banks enable mechanistic interpretability, revealing linguistic structures—such as paragraph lengths—and identifying redundant attention heads for model pruning. With only a few additional parameters, GPA enhances efficiency without sacrificing performance. Our work bridges kernel methods and Transformers, providing a theoretical lens for attention while delivering practical gains in performance and interpretability. We pave the way for scalable, interpretable NLP models, with implications for optimizing large-scale Transformers and understanding their inner workings.

1 Introduction

The Transformer model (Vaswani et al., 2017) has revolutionized artificial intelligence, and has become a key foundational architecture across diverse domains such as NLP (Kalyan et al., 2021), computer vision (Khan et al., 2022; Han et al., 2022), speech recognition (Gulati et al., 2020), computational biology (Zhang et al., 2023), and more. Nevertheless, Transformers remain more of a heuristic than a formal scientific framework. An underlying theory explaining not just how, but why they work has remained elusive, but such a theory is, arguably, essential for predicting safety, reliability, and alignment (Bereska and Gavves, 2024). Theoretical models are useful at several levels. They provide intuition, but more importantly, they establish a framework for analyzing errors and are a springboard for the invention of new algorithms. The objective of this work is to present a modified self-attention model that both improves performance and provides a basis for interpreting characteristics seen in results when used in inference.

2 Methodology

2.1 Theory

The methodology used in this work builds on Nadaraya-Watson (NW) regression (Nadaraya, 1964; Watson, 1964), which uses a set of observed points, $\{x_i, y_i\}$, $i = 1, \ldots, N$, and a kernel function, K_h, to estimate the value of y at any new point, x. The estimate, $\hat{y}$, is computed as a normalized-weighted, shifted sum of the kernel function:

$$\hat{y} = \mathrm{NW}(x) = \sum_{i=1}^{N} \left[\frac{K_h(x - x_i)}{\sum_{i=1}^{N} K_h(x - x_i)} \right] y_i \quad (1)$$

In this expression, the function, K_h, is centered around each of the x_i and weighted by the corresponding y_i. This shifted and normalized weighted sum forms the regression function. The shape of K_h is typically a symmetric, Gaussian-like curve whose width is controlled by a parameter h. Figure 1 illustrates a simple 1D example.

When self-attention is implemented using rotary positional embeddings (RoPE) (Su et al., 2024), its form is the same as NW regression

$$\mathrm{Att}(x_n) = \sum_{i=1}^{N} \left[\frac{\exp(\frac{x_n^T Q^T \Theta^T \Theta K x_i}{\sqrt{d}})}{\sum_{k=1}^{N} \exp(\frac{x_n^T Q^T \Theta^T \Theta K x_k}{\sqrt{d}})} \right] V x_i \quad (2)$$

Here the x_i are vectors in $\mathcal{R}^d$ and $x_n \in \mathcal{R}^d$ is the target vector. The matrices Q, K, and V are learned parameters, and RoPE is a deterministic sparse matrix Θ that operates on the query, Qx_n,

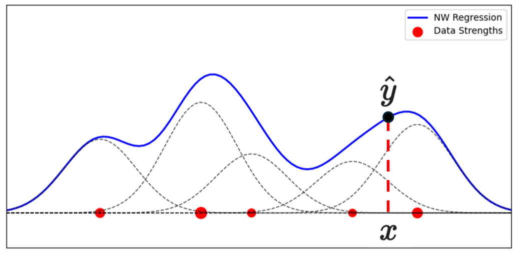

Figure 1: Illustrating Nadaraya-Watson regression. The resulting regression function, shown as a blue curve, is the weighted sum of shifted kernel functions, each shown as a black, dashed curve. The data locations, x_i, are represented by the red dots on the horizontal axis, and their values, y_i are represented by the size of the dots. The regression for a new data point, x, is shown on the regression curve as $\hat{y}$.

and key, Kx_i vectors. In Equation 2, this manifests as the block $\Theta^T\Theta$. The structure of Θ is block diagonal, where each block is a 2D rotation matrix. The angles of rotation increase as a function of index and position. One of the key characteristics of RoPE is that $\Theta^T\Theta$ is a function of the indicial distance between embeddings (Su et al., 2024). Attention, and NW regression, both form normalized weighted sums dependent on relative distances. Thus, attention can be interpreted as a proper kernel function centered around each x_i (Tsai et al., 2019). Although attention is not symmetric, asymmetric kernels have been formalized in both theoretical frameworks and practical applications, and are useful for modeling conditional probabilities and directed graphs (He et al., 2023b,a; Wu et al., 2010).

2.2 Kernel Modeling

A useful characteristic of kernel functions is that they can be combined through summation or multiplication, and the result remains a valid kernel (Aronszajn, 1950). This is useful for modeling. RoPE is thought to implicitly embody the decaying periodic correlations known to be part of the structure of language (Barbero et al., 2024), and the goal of this section is to redesign attention, enriching RoPE with kernel functions designed to capture these features more precisely. We begin by defining two kernel functions. The first, P_k, models periodicity, and the second, D_k, exponential decay:

$$P_k(x_n, x_i) = \exp\left\{-2\alpha_k^2 \sin^2\left(\frac{|n-i|}{\tau_k}\right)\right\} \quad (3)$$

$$D_k(x_n, x_i) = \sigma_k^2 \exp\left\{-\frac{|n-i|}{l_k}\right\} \quad (4)$$

Each kernel is an explicit function of the indicial distance between the target and context vectors, x_n and x_i, respectively. P_k is a function of two learnable parameters, α_k and τ_k, where the former controls amplitude and the latter period. D_k depends on the learnable parameters, σ_k and l_k, where the former is the strength of the term and the latter is a time constant or decay width parameter. The two kernels can be multiplied to model decaying periodicity, and a complex kernel function can be formed as the sum over a bank of M such kernels:

$$G(x_n, x_i) = \sum_{m=1}^{M} D_m(x_n, x_i) P_m(x_n, x_i) \quad (5)$$

Finally, the expression in Equation 5 can be combined with that for attention from Equation 2 yielding a new kernel function, GPA(x_N), given by:

$$\sum_{i=1}^{N} \text{softmax}\left[G(x_n, x_i) + \frac{x_n^T Q^T \Theta^T \Theta K x_i}{\sqrt{d}}\right] V x_i \quad (6)$$

Because kernel functions are often used to represent Gaussian stochastic processes (Wilson and Adams, 2013), we call this model Gaussian Process Attention (GPA). In comparison with standard RoPE, it introduces $4M$ additional learnable parameters per attention head.

2.3 Experimental Setup

Our study utilizes a compact GPT Transformer architecture composed of four layers, each featuring four attention heads. The experimental dataset comprises the complete works of Charles Dickens, sourced from Project Gutenberg (Dickens, 2018), and employs a character-level tokenization method as described in (Banar et al., 2020). This corpus includes approximately 13 million characters and a vocabulary size of 93 tokens. By adopting this approach, we simplify the language preprocessing typically involved in training Transformer-based language models. Additionally, this method avoids the need to replace infrequent tokens with placeholders such as <UNK>, resulting in a concise and well-defined vocabulary.

Our model incorporates the standard components found in Transformer blocks, including layer normalization, linear projection layers, multilayer

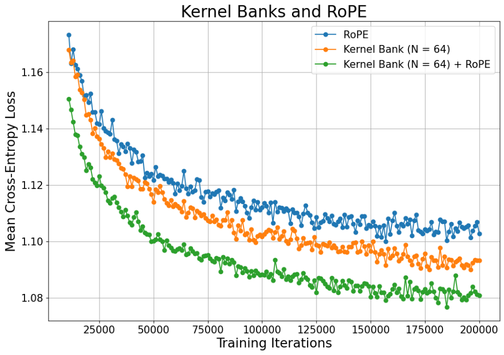

Figure 2: Comparison of validation curves during training for three experiments, each run for 200k iterations. The blue curve is the MCE loss for the baseline implementation of attention with Rotary Positional Embeddings (RoPE), orange is for the GPA kernel bank excluding RoPE, and green is GPA combined with RoPE.

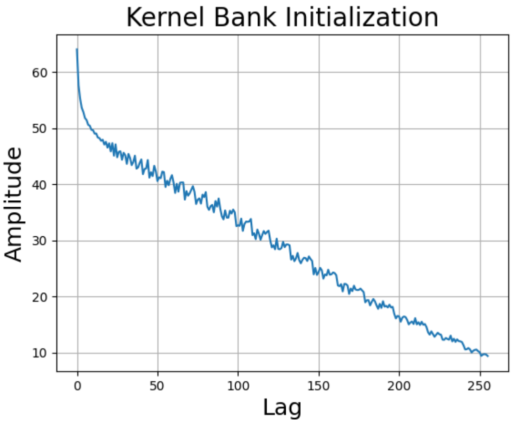

Figure 3: Functional shape of a kernel bank at initialization.

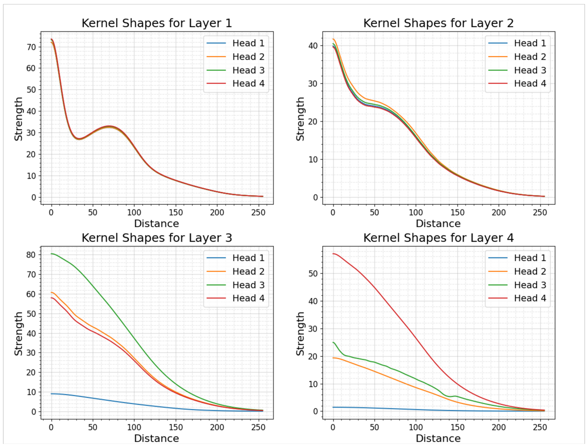

Figure 4: Shapes of the 16 trained kernel banks after 200k training iterations.

perceptrons, as well as embedding and unembedding layers.[1] For our experiments, we set the context window length to 256 tokens and use an embedding dimension of 512. Model performance is evaluated using the mean cross-entropy (MCE) loss on the validation set.

3 Experimental Results

Three experiments were run to evaluate the kernel models of the previous sections, and the results are shown in Figure 2. Each curve in the figure represents the MCE loss during training as applied to the validation data. The experiments consist of 200k gradient update iterations, with a batch size of 256 (equivalently 4 epochs). The data split is 90% for training and 10% for validation. The baseline experiment, represented by the blue curve, is the MCE loss for the RoPE implementation, as specified in Equation 2, of the GPT architecture. The green curve is the MCE loss for our GPA formulation as described by Equation 6, where the bank is constructed from $M = 64$ decaying periodic kernels. The orange curve is an additional experiment that implements the GPA kernel bank and does not use RoPE, nor any other positional information other than that provided by the formulation of the kernel bank. The examples show that the kernel bank works as well as RoPE in modeling relative positional information, while also capturing additional predictive characteristics of the data. Of the three, the best performing model is the composition of the two techniques. This suggests that the kernel bank and RoPE capture complementary characteristics of the data.

3.1 Kernel Bank Shapes

As detailed in Section 2.3, our model implements four layers with four attention heads per layer. Each of the 16 kernel banks consists of a sum of 64 learned decaying periodic functions. Figure 3 shows the shape of the initialized kernel banks, where $\alpha_k = 1$, $\sigma_k = 1$, and $l_k = 150$ for all k, and the τ_k take 64 evenly space values in the interval $[4, 192]$. Figure 4 shows the functional form of each of the 16 kernel banks after 200k training iterations. The figure has a number of interesting features. The shapes of the four kernel banks in the first layer (shown in the upper left) are almost identical, and each has a prominent bump at lag 70. The second layer (upper right) is similar to the

[1] See https://transformer-circuits.pub/2021/framework/

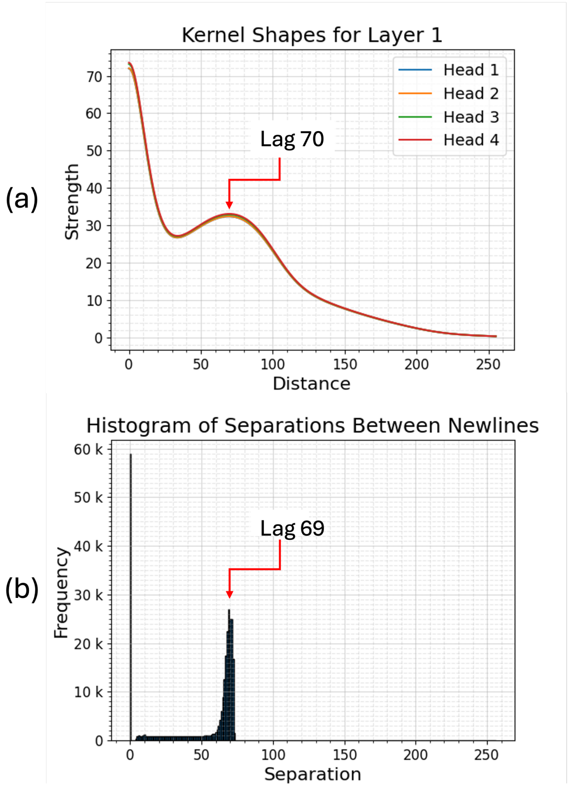

Figure 5: (a) Shapes of the trained kernel banks for the first layer of the model, and (b) Histogram of the separation between newlines in the data

first, showing only small differences between the four heads. The third and fourth layers, however, more clearly differentiate the shapes between the heads. Notably, the strengths of the banks begin to vary, and in the fourth layer, one of the kernel banks is effectively zero. This suggests that the final layer of this model could discard one of the attention heads and its associated MLP without any loss of performance in inference. This would result in savings in both compute and memory.

3.2 Analyzing the Bump

An examination of the corpus data suggests an explanation for the location of the kernel bump seen in the first layer of Figure 4. Our conjecture is that the bump corresponds to the average length of paragraphs, represented as a double newline. To test this, we computed the histogram of separations between newlines in the corpus, and the result is illustrated in Figure 5. The histogram of newline separations (part (b) of the figure) shows a peak at lag 1 and another at lag 69. The first peak is due to the fact that new paragraphs in the corpus are the result of two successive newlines, having a lag of 1. The peak at lag 69 seems to confirm the conjecture that the bump in the kernel shapes at lag 70 (shown in part (a) of the figure) is capturing this characteristic. An additional histogram (not shown), computed from 50,000 characters generated by the trained model, puts the peak at precisely lag 70, further supporting the conjecture.

4 Discussion and Future Work

This paper demonstrates the potential of kernel functions to better model language, improve performance, and to serve as a foundation for experiments in interpretability. Our results show that kernel functions capture important predictive characteristics in the data, improving the performance of RoPE. The computational cost of this additional predictive power is nominal, adding just 4,096 parameters to a GPT model consisting of 13.8m weights.

In addition to improved performance, the kernel banks provide new opportunities for mechanistic interpretability. We studied a notable characteristic of a kernel function that correlates to an interpretable feature in the data. We also observed that one of the trained kernel functions was uniformly zero, suggesting that its attention head was redundant. This is a valuable insight because it means that this head and its MLP can be removed for both training and inference, and in so doing, reduce associated computational and memory costs.

5 Limitations

The results presented in this paper seem promising, but are for a small corpus and a small GPT model. Experiments with a larger corpus (for example, Wikipedia or Fineweb (Penedo et al., 2024)) would validate the kernel bank efficacy for a more consequential dataset. The character-based tokenization strategy used for this paper was useful, as it allowed us to circumvent the many design and engineering questions related to vocabulary size that come with more sophisticated tokenization schemes such as WordPiece (Schuster and Nakajima, 2012) or byte-pair encoding (Sennrich et al., 2016). The tradeoff is that character-based tokenization loses much of the semantic information derived from words. Finally, the kernel bank models need to be tested in downstream applications. Doing so would provide additional insight into their strengths, weaknesses, and capabilities.

Acknowledgments

Removed for anonymity.

References

Nachman Aronszajn. 1950. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404.

Nikolay Banar, Walter Daelemans, and Mike Kestemont. 2020. Character-level transformer-based neural machine translation. In *Proceedings of the 4th International Conference on Natural Language Processing and Information Retrieval*, pages 149–156, New York, NY, USA. Association for Computing Machinery.

Federico Barbero, Alex Vitvitskyi, Christos Perivolaropoulos, Razvan Pascanu, and Petar Veličković. 2024. Round and round we go! what makes rotary positional encodings useful? *arXiv preprint arXiv:2410.06205*.

Leonard Bereska and Stratis Gavves. 2024. Mechanistic interpretability for AI safety - a review. *Transactions on Machine Learning Research*.

Charles Dickens. 2018. Index of the project gutenberg works of Charles Dickens. https://www.gutenberg.org/ebooks/58157. Public domain. Accessed: 2025-05-15.

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and 1 others. 2020. Conformer: Convolution-augmented transformer for speech recognition. In *Proc. Interspeech 2020*, pages 5036–5040.

Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, and 1 others. 2022. A survey on vision transformer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(1):87–110.

Fan He, Mingzhen He, Lei Shi, Xiaolin Huang, and Johan AK Suykens. 2023a. Enhancing kernel flexibility via learning asymmetric locally-adaptive kernels. *arXiv e-prints*, pages arXiv–2310.

Mingzhen He, Fan He, Lei Shi, Xiaolin Huang, and Johan AK Suykens. 2023b. Learning with asymmetric kernels: Least squares and feature interpretation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(8):10044–10054.

Katikapalli Subramanyam Kalyan, Ajit Rajasekharan, and Sivanesan Sangeetha. 2021. AMMUS: A survey of transformer-based pretrained models in natural language processing. *Language*, 4.

Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. 2022. Transformers in vision: A survey. *ACM Computing Surveys (CSUR)*, 54(10s):1–41.

Elizbar A Nadaraya. 1964. On estimating regression. *Theory of Probability & Its Applications*, 9(1):141–142.

Guilherme Penedo, Hynek Kydlíček, Anton Lozhkov, Margaret Mitchell, Colin A Raffel, Leandro Von Werra, Thomas Wolf, and 1 others. 2024. The fineweb datasets: Decanting the web for the finest text data at scale. *Advances in Neural Information Processing Systems*, 37:30811–30849.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and Korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152, Piscataway, NJ, USA. IEEE, Institute of Electrical and Electronics Engineers.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, page 1715. Association for Computational Linguistics.

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

Yao-Hung Hubert Tsai, Shaojie Bai, Makoto Yamada, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2019. Transformer dissection: An unified understanding for transformer's attention via the lens of kernel. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*, 30:5998–6008.

Geoffrey S Watson. 1964. Smooth regression analysis. *Sankhyā: The Indian Journal of Statistics, Series A*, 26(4):359–372.

Andrew Wilson and Ryan Adams. 2013. Gaussian process kernels for pattern discovery and extrapolation. In *International Conference on Machine Learning*, pages 1067–1075. PMLR.

Wei Wu, Jun Xu, Hang Li, and Satoshi Oyama. 2010. Asymmetric kernel learning. Technical report, Microsoft Research. Technical Report.

Shuang Zhang, Rui Fan, Yuti Liu, Shuang Chen, Qiao Liu, and Wanwen Zeng. 2023. Applications of transformer-based language models in bioinformatics: a survey. *Bioinformatics Advances*, 3(1):vbad001.