2025 International Conference on Intelligent Computing
July 26-29, Ningbo, China
https://www.ic-icc.cn/2025/index.php

T-Attention: Optimizing Attention Computation Using Temporal Parameter Time

Yichen Yang and Hongxu Hou✉ and Wei Chen

College of Computer Science, Inner Mongolia University, Hohhot, China

Abstract. The attention mechanism exhibits remarkable capability in processing sequential data; however, its computational complexity scales quadratically with sequence length, resulting in significant resource demands. Numerous studies have achieved substantial success in leveraging sparse matrices to reduce the computational burden of dot-product operations, thereby improving both the computational efficiency and accuracy of models. Nevertheless, the question remains: can we further optimize these computations? In this paper, we introduce a novel approach based on function projection, integrating a restructured word embedding technique with the attention mechanism to alleviate computational overhead. We first validate the theoretical efficacy of designing word embedding using parametric equations and demonstrate the effectiveness of our proposed embedding method. Subsequently, we conduct experiments across a variety of basis functions, illustrating that our approach affords greater flexibility in parameter selection while effectively reducing computational costs. Compared to state-of-the-art attention-based models, our method achieves an average reduction of 60% in parameter count, 75% in FLOPs, and 63.4% in inference time under specific parameters.

Keywords: Attention, Parametric Equations, Fourier Series, Function Projection.

1 Introduction

Since the introduction of the attention mechanism, its effectiveness has been demonstrated on multiple levels and has continued to play a crucial role in the development of subsequent mainstream models [1]. It has been widely applied in both computer vision [2] and NLP [3]. Additionally, it has achieved outstanding results in emerging fields such as graphs [4] and reinforcement learning [5]. The widespread adoption of models such as GPT [6][7] further demonstrates the strong representational power of the attention mechanism.
Since the introduction of the attention mechanism, computational power stacking has become a core paradigm in machine learning. However, the computational complexity of attention mechanisms increases exponentially when handling long sequences, prompting researchers to explore methods that can reduce attention computation while improving model performance. Sparse attention [8] is a mainstream

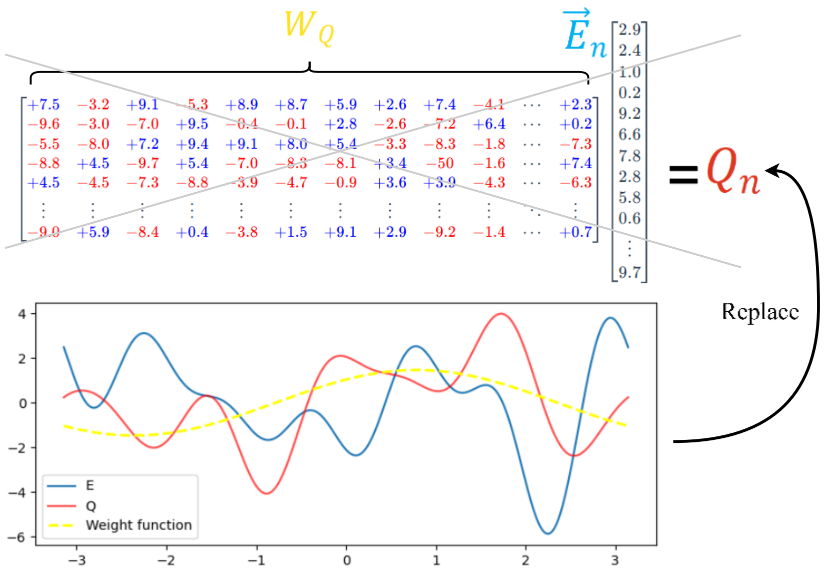

Fig. 1. We replaced matrix projection with function projection, where a projection function with only two parameters can provide sufficiently complex linear transformations.

solution, and some researchers have also introduced additional mechanisms to optimize the algorithm [9-15].

However, we believe that computational optimization for the attention mechanism can be further improved. LM-Steer [16] pointed out that attention should be focused on vectors, and we argue that the concept of "continuity" provides a valuable perspective. Differential equations [17], and PINNs [18] have already demonstrated the strong capability of models in solving problems with continuity (non-classification and non-generation tasks). However, such models perform poorly on traditional classification problems because solving differential equations inherently introduces a time component, which is irrelevant to classification tasks.

When analyzing the process of the attention mechanism, the concept of time can be meaningfully introduced. Therefore, we propose a time-based attention mechanism and a time-aware word embedding method, leveraging Fourier series and parametric equations to reduce the computational cost of attention mechanisms. The overall structure of our method is illustrated in Figure 1. Our approach replaces traditional matrix projections with function projections.

Our contributions are as follows:

1.We propose a novel word embedding approach designed to enhance the training efficiency of deep networks.

2.Our method enables faster computation of contextual influences on vectors, improving efficiency in attention-based models.

3.We conducted experiments with various basis functions to extend the applicability of our method.

2 Related Work

2.1 Sparse Attention

Sparse Attention [19] reduce computational complexity by introducing sparse attention patterns, making the Transformer more efficient for long sequences. Blockwise Attention [20] divides the sequence into multiple blocks and computes attention within or between blocks, reducing computational costs and improving parallelism. Linformer [21] compresses the attention matrix into a fixed size using low-rank projection, reducing both computation and memory requirements, making the Transformer more suitable for long sequences. Reformer [22] approximates self-attention using locality-sensitive hashing (LSH) and incorporates reversible residual networks, significantly lowering the computational and storage overhead of the Transformer. Ring Attention [23] computes attention based on a ring structure, focusing only on local and partial global information, enhancing the model’ s capability for long-sequence modeling while reducing computational cost. Longformer[24] combines local windowed attention with global attention mechanisms, improving the Transformer’ s ability to process long texts while maintaining computational efficiency. Adaptive Attention Span [25] dynamically adjusts the attention window size, allowing the model to flexibly select the attention range based on requirements, thereby improving both computational efficiency and expressive capacity.

in recent years, methods such as Agent Attention [26] and Multimodal Attention Bottlenecks [27] have continued to advance the development of attention mechanisms. NSA integrates multiple sparse matrix methods and employs gated units to achieve a more efficient sparse attention mechanism. However, traditional methods cannot avoid the substantial computational load brought by matrix projection, with most of the computational loss concentrated in this part. We address this by proposing a new projection mechanism to comprehensively alleviate the pressure in this area.

2.2 KV Cache

In optimizing attention, the KV cache method has achieved remarkable results in recent years [28]. Minicache analyzes the redundancy in KV cache usage for large models [29]. Keyformer reduces KV cache storage through key token selection [30], while KVquant integrates quantization concepts to enable ultra-long-context inference in large models [31]. "KV cache is 1 bit per channel" [32] introduces coupled quantization for KV cache as an alternative to channel-based caching methods. It is important to note that KV cache methods also pose new challenges for hardware, and the combination of GQA [33] and NSA provides an efficient solution. The core idea of the KV cache method is to reduce the number of matrix projection operations, but it still cannot avoid the computational overhead caused by the remaining projection operations.

2.3 Continuity in Machine Learning

Fourier series has been widely applied in the field of machine learning. In its early stages, Fourier transform was primarily used for feature extraction. Over time, it has been increasingly integrated with machine learning techniques. Several notable studies have demonstrated its effectiveness. Some researchers integrates Fourier series with positional encoding to enhance spatial representations [34], and incorporates Fourier series into PINNs, improving their ability to solve multi-scale partial differential equations [35]. Some researchers applies Fourier series to quantum machine learning, leveraging its expressive power [36], and integrates Fourier series into MLPs, enabling efficient time series forecasting [37]. AFGAN replaces conventional adversarial samples with Fourier-based features, enhancing generative model performance [38].

In machine learning, the relationship with differential equations is often explored in the context of solving partial differential equations (PDEs). However, some researchers argue that differential equations and machine learning are deeply interconnected. ODEnet [42] bridges differential equations with neural networks, providing a strong theoretical foundation and improved interpretability for neural architectures. Some researchers discusses modeling high-dimensional nonlinear functions using continuous dynamical systems [44]. Some suggests that many efficient convolutional neural networks can be interpreted as differential equations [45]. PolyNet [46] enhances the expressive power of neural modules by introducing the PolyInception module. Neural ODEs [47] demonstrates that the macro-architecture of FractalNet can be understood through the well-known Runge-Kutta method in numerical analysis. FFJORD [48] further explores the role of differential equations in flow-based models. These research directions and findings indicate that differential equations can be effectively integrated with deep neural networks. Their mathematical and theoretical foundations provide valuable insights for constructing more robust and interpretable machine learning systems, this provides a theoretical basis for our approach.

3 Method

Our method is divided into three parts: Section 3.1 introduces the concept of parametric equations and their mathematical significance; Section 3.2 presents our reconstructed word embedding method, which serves as a prerequisite for our attention mechanism; Section 3.3 explains how our attention mechanism is constructed.

3.1 Preview

Parametric equations are a mathematical approach used to represent curves or surfaces through parameterized variables. Compared to traditional explicit or implicit functions, parametric equations offer significant advantages in describing complex curves, motion trajectories, and high-dimensional geometric structures.

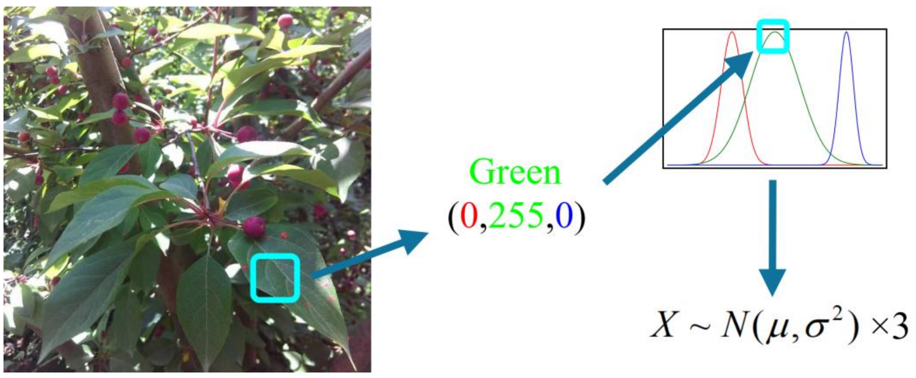

Fig. 2. The color of a single pixel corresponding to its value in a space defined by three normal functions.

In a two-dimensional space, if there exists a parameter $t \in I$(where I is some interval), a planar curve can be expressed as:

$$x = f(t), y = g(t) \tag{1}$$

This curve is then defined by the parametric equations $(x, y) = (f(t), g(t))$, where t is referred to as the parameter. To meet the value requirements of the subsequent word embedding section, the function must be periodic, related to Equations 5 and 6. The choice of function is flexible, and in our experimental section, we used Fourier series as a component of the parametric equation for word embedding. This representation effectively captures the dynamic properties of the curve, such as motion trajectory and directional information.

The concept of parametric equations allows us to associate the notion of time with vectors. For a vector $x,\dots,y$, its elements can be interpreted as the values of n equations evaluated at a given time t. We propose that these n equations can take various forms, including discrete equations, Fourier series, or differential equations, as illustrated in Fig 2.

Our theory originates from the RGB color space, where the underlying principle is a two-dimensional function of wavelength and transmittance. Under different observational methods, the RGB model consistently relies on three functions, each representing the transmittance at a specific wavelength. In computer science, these three functions are abstracted as approximately normal distributions and reconstructed as a two-dimensional function of output and channel values. Despite this transformation, the fundamental characteristics remain intact, enabling highly effective performance in image computation.

We first demonstrate the general applicability of the RGB logic in the field of image processing. The experimental setup and results can be found in Section 4.1. Our approach involves modifying the three underlying functions that govern RGB representation through various algorithms. We then evaluate whether the modified images can still be accurately recognized by machine learning models when compared to the original images. To achieve this, we employ the color transformation matrix method, which allows us to systematically alter the RGB functions while preserving the structural integrity of the image:

$$\begin{bmatrix} R' \\ G' \\ B' \end{bmatrix} = \begin{bmatrix} 0.5 & \cdots & 0.2 \\ \vdots & \ddots & \vdots \\ 0.3 & \cdots & 0.8 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{2}$$

This computational logic modifies the three functions in the RGB space. When chosen appropriately, the color transformation matrix functions similarly to a convolution kernel, potentially leading to a slight improvement in model accuracy. Its differential equation form is given by:

$$\frac{d}{dt}\begin{bmatrix} R(t) \\ G(t) \\ B(t) \end{bmatrix} = M \cdot \begin{bmatrix} R(t) \\ G(t) \\ B(t) \end{bmatrix} \tag{3}$$

where M is the transformation matrix that governs the evolution of the RGB functions over time. This equation describes the dynamical evolution of a three-dimensional state variable. The nature of the solution depends on the eigenvalues of the transformation matrix M.

For rigor, we explored more complex oscillatory behaviors by using higher-dimensional matrices or introducing damping and coupling terms. Specifically, we considered Damped oscillations, Forced oscillations, Coupled oscillatory systems. From this, we infer that not only can colors be represented by parametric equations, but any form of data can fundamentally be expressed in this way. For complex systems, we can extend the number of parametric equations or increase the complexity of individual functions to achieve a more accurate representation.

3.2 Time-Aware Word Embedding

Researchers often refine vector representations or apply further transformations to improve performance. The method of word embedding fundamentally determines how the attention mechanism operates**.** Current embedding approaches convert data into vectors, which is an inherent characteristic that cannot be altered. However, as described in Section 3.1, we propose embedding vectors as the values of a parametric equation at a specific time point.

Once word vectors enter the model, they pass through layers such as attention mechanisms, where they undergo dot products, scalar multiplications, and accumulations—

which collectively contribute to computational overhead. Our goal is to simplify the entire computational process into a single unified logic.

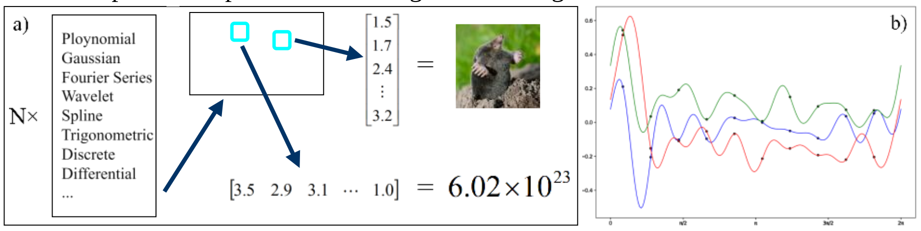

Fig. 3. Word embedding space constructed by parametric equations

For example, consider determining the meaning of the word "mole". The model analyzes contextual information to infer whether "mole" refers to an animal, a chemical unit, or another meaning. Initially, the word is represented by an embedding vector E_1, which, after complex computations, results in a fixed set of output vectors $(E_2, E_3, \ldots)$. Instead of treating E_1 as a standalone vector, we reinterpret it as a set of parametric equations $(f_1, f_2, \ldots, f_n)$ evaluated at time t_1, as illustrated in Fig 3a. When f_1 reaches its maximum, the vector possibly represents E_1. When f_2 reaches its maximum, the vector possibly represents E_2. Collectively, the n parametric equations define the parameter space of the word "mole".

Thus, when constructing word embedding, we introduce a vector transformation rule on top of the existing vector space. By employing parametric equations, we impose constraints on the possible transformations of vectors, thereby preventing unnecessary computations in later stages. Each vector can be represented as:

$$\overrightarrow{E_n} = [f_1^n(t), f_2^n(t), \ldots, f_d^n(t)] \tag{4}$$

n represents the vector identifier, and d denotes the number of dimensions, which corresponds to the number of parametric equations. In the RGB color space, a set of three well-defined functions forming a parametric equation can represent 16,777,216 different colors. Clearly, this number is more than sufficient for capturing the semantics of a word. However, the directionality of representation is limited. Therefore, in NLP tasks, a higher-dimensional space is essential to accurately capture the diverse characteristics of word vectors.

After completing the original word embedding task, we collect all vectors that follow the same transformation rules and assign each word a new space, which we refer to as the "temporal space." The domain of this space is defined as $[0, 2\pi]$, and each vector is evenly distributed within this domain, as shown in Fig 3b. To fit these data points, we use Fourier series as the fitting function. In principle, there are countless possible ways to fit the functions within the parametric equations. We could even employ random methods, nonlinear functions, or discrete functions. However, all selected functions must be periodic, and the data points at both ends must be adjusted to ensure periodicity, this condition is a prerequisite for **T-attention**.

The advantage of designing word embeddings in this way is that traditional word embedding methods assign a unique vector to each word. However, in reality, any vector can be obtained through a linear transformation of other vectors. For example, a word's base form, past tense, and adjective form often exhibit high similarity, and the transformation process between them is fixed. This suggests that a special set of vectors can belong to the same system, represented as a parametric equation. By expressing vectors in the form $v = f(t)$, we capture the intrinsic relationships between word variations while reducing redundant computations.

3.3 T-attention

Under the concept of higher-dimensional parametric equations, vectors can represent more meanings. The final output value of a vector at the end of the model's processing is determined by time. After modifying the original attention mechanism, our revised formula is:

$$\Delta t = Att(\frac{f_q^n(t) f_k^n(t)}{\sqrt{d_k}}) f_v^n(t) \tag{5}$$

$$pre = f^n(t + \Delta t) \tag{6}$$

Where f_q^n and others are functions after projection, used to replace the original K, Q, V projection matrix. The parameter t is related to the word embedding method mentioned earlier, where each word vector corresponds to a unique parameter t in the embedding function. Δt represents the distance on the function between the original word vector and another word vector after attention. When the dimension of the function projection matrix is smaller than that of the input function, it manifests as a downsampling process.

The above formula is used to calculate the time parameter t. Alternatively, we can directly compute the projection of the function to update word embeddings and train the word model:

$$f_{new} = Att(\frac{f_q^n f_k^n}{\sqrt{d_k}}) f_v^n \tag{7}$$

Thus, under this method, there is no need to update the large K, Q, V matrices. Instead, we only need to define the number of basis functions and update the basis functions themselves. For a single basis function $sin(nx)$, there is only one trainable parameter, significantly reducing the computational complexity.

Compared to traditional matrix projection, the time complexity of function projection depends on the choice of basis functions, and the computer requires additional time to compute these basis functions. However, our experiments in Table 4 demonstrate that this extra cost does not cause our method to lag behind traditional methods, except in cases with certain basis functions that are difficult to compute, where performance is suboptimal.

4 Experiment

Color transformation and oscillatory systems. The underlying RGB logic in image processing is often implicit in machine learning, making it necessary to formally establish its presence and demonstrate that modifying it can yield positive effects. The values given in Equation (2) represent just one randomized instance, and we have tested various color transformation matrices to validate that such transformations do not negatively impact the final results.

Moreover, the color transformation matrix empirically confirms the existence of the RGB color space and demonstrates that this space can be represented as a parametric equation and transformed into a vector form. It is important to note that we do not perform transformations on individual pixels but rather apply global color transformations to the entire image. This bears similarity to convolution kernel computations, as both operate over structured spatial information. Thus, although color transformation matrices can enhance image recognition performance, the selection of transformation values should be constrained within a reasonable range rather than being overly flexible. We conducted experiments on the CIFAR-10 dataset, and the results are presented in Table 1.

From the table data, we can observe that methods applying color transformations to the entire image generally do not affect the quality of the image in model training, and in most cases, they lead to an improvement in metrics. Our experiments were conducted on a lightweight model (1500 parameters). The reason for not using the standard ResNet model was to test the logical similarities between pure convolution and our method. This experiment aimed to validate the theoretical foundation (**The information of an image can be obtained by sampling a set of parametric equations, and similarly, text can be processed in the same way. However, the parametric equations underlying text require further exploration.**) and is unrelated to subsequent experiments.

However, the transformations provided by the color transformation matrix are limited, as their dynamic changes are globally monotonic. Therefore, we further experimented with the effects of damped oscillations, forced oscillations, and coupled oscillations on the data. Among these, forced oscillations allowed for more parameter adjustments. As shown in Table 2, the model maintained good performance under various oscillation types and parameter influences. The above systems are all linear. To explore further, we tested the effects of nonlinear systems, such as the Lorenz attractor and the Van der Pol oscillator. However, nonlinear systems did not yield satisfactory experimental results.

The parameters for damped oscillation are K, c, t (damping constant, damping coefficient, time). The parameters for forced oscillation are K, c, A, Ω, t (spring constant, damping coefficient, amplitude of the forcing, frequency of the forcing, time). The parameters for coupled oscillation are K, c, t (coupling constant, damping coefficient, time).

Table 1. We tested the impact of individual transformation matrices on model performance, including random matrices, random matrices within specified intervals, and classic filters, demonstrating that color transformations rarely have a negative impact on the image itself. The purpose of this experiment was to prove that the RGB color selection logic is determined by a function, and this function, when freely valued, does not affect the model's judgment of the image.

Setting	Acc	Setting	Avg.Acc	Setting	Acc
No change	54.10	Random [0~2]	57.36	Sobel-X	57.14
Random set	61.32	Random [1~2]	56.03	Sobel-Y	36.78
Random set	60.45	Random [-1~1]	62.99	Laplacian	56.04
Random set	60.37	Random [-2~2]	59.52	Sharpening	56.23
Random set	58.59	Random [-1~0]	10.00	Blur	58.89
Random set	59.51	Random [-2~0]	10.00	Gaussian Blur	58.34
Random set	60.82	Random [0~5]	55.09	High-pass Fliter	55.10
Random set	60.94	Random [0~10]	50.21	Low-pass Fliter	64.60
Random set	59.37	Random [-10~10]	56.62	Motion Blur	58.74
Random set	60.72	Random [0~0.01]	63.66	Edge enhance	57.80

Table 2. Non-monotonic oscillatory systems. The parameters in the table represent the variables of the oscillatory system.

Setting	Acc	Setting	Acc	Setting	Acc	Setting	Acc
Damped (1.0,0.5,0.1)	63.62	Damped (1.0,0.5,0.1)	62.92	Forced (1.0,0.2,0.2,2.0,0.1)	62.53	coupled (1.0,0.5,0.1)	62.00
Damped (1.0,0.5,0.1)	62.89	Forced (1.0,0.2,0.2,2.0,0.1)	62.48	Forced (1.0,0.2,0.2,2.0,0.1)	62.89	coupled (1.0,0.5,0.1)	62.18
Damped (1.0,0.5,0.1)	63.24	Forced (1.0,0.2,0.2,2.0,0.1)	55.69	Forced (1.0,0.2,0.2,2.0,0.1)	63.73	coupled (1.0,0.5,0.1)	63.10
Damped (1.0,0.5,0.1)	62.86	Forced (1.0,0.2,0.2,2.0,0.1)	44.89	Forced (1.0,0.2,0.2,2.0,0.1)	62.91	coupled (1.0,0.5,0.1)	62.01
Damped (1.0,0.5,0.1)	62.45	Forced (1.0,0.2,0.2,2.0,0.1)	62.78	Forced (1.0,0.2,0.2,2.0,0.1)	61.97	coupled (1.0,0.5,0.1)	60.36
Damped (1.0,0.5,0.1)	62.41	Forced (1.0,0.2,0.2,2.0,0.1)	63.31	Forced (1.0,0.2,0.2,2.0,0.1)	64.94	coupled (1.0,0.5,0.1)	62.18

Word Embedding Comparison Experiment We conducted a comparative study with static word embeddings, including Word2Vec (CBOW / Skip-gram), GloVe, and FastText, as well as contextualized dynamic word embeddings, such as BERT, ELMo, and context2vec. The comparison was performed across multiple datasets. To validate the adaptability of our method, we applied our word embedding approach to

Table 3. Word embedding comparison experiment. This experiment verifies that our word embedding method can be applied to any word embedding approach to replicate its embedding representation. It cannot surpass the performance of other methods but can reproduce their results.

	Methods	WS353	WS353S	MEN	SimLex999	RG-65
Static	Skip-gram	61.0	68.9	67.0	34.9	75.2
	CBOW	62.7	70.7	68.6	38.0	72.7
	Glove	54.2	64.3	68.3	31.6	61.8
	FASTTEXT	68.3	74.6	74.8	38.2	80.8
	Deps	60.6	73.1	60.5	39.6	77.1
	our+best	**68.3**	**74.6**	**58.6**	**39.6**	**80.8**
Contextualized	ELMo	45.5	62.1	57.2	40.6	60.9
	GPT2	30.7	31.4	26.2	26.4	10.6
	BERT	24.0	31.0	22.0	13.4	18.5
	XLNet	62.8	69.8	61.7	49.0	63.4
	our+best	**62.8**	**69.8**	**61.7**	**49.0**	**63.4**
Context-to-Vec	Context-LSTM	63.5	66.6	66.4	39.3	72.6
	SynGCN	60.9	73.2	71.0	45.5	79.6
	Bert+skip-gram	72.8	75.3	76.2	49.4	78.6
	Graph-Retro	78.9	77.0	77.9	55.2	85.1
	Our+best	**78.9**	**77.0**	**77.9**	**55.2**	**85.1**

the best-performing models in each of the three classification categories. The results demonstrated that our method can be seamlessly integrated into any existing word embedding approach without compromising embedding quality.

It is important to note that in our word embedding process, all functions within the parametric equations used for each vector group were uniformly fitted using Fourier series. We conduct experiments on the Word Similarity task, and the dataset comes from [39-42]. The experimental results are shown in Table 3.

Performance of T-Attention In our method, the number of parameters is determined by the number of basis functions, computational complexity is influenced by the type of basis functions, and the relationship between the number of basis functions and the original function determines whether upsampling or downsampling occurs. By designing appropriate basis functions, we can adjust the model's performance. To validate this, we conducted experiments under various specialized designs.
In our experiments, we selected four baselines for comparison. Full-att: This does not use a sparse attention mechanism. Sparse_A: Cannot optimize parameters but improves PPL. Sparse_B: Significantly reduces the number of parameters but introduces sparse matrix computations, which slow down training speed. NSA : a me-

Table 4. The first six rows of experiments represent six typical matrix projection methods (baseline), while the remaining experiments involve function projection. The performance of the four evaluation metrics depends on the choice of basis functions and their parameters.

Method	Appropriate	Para/M	Time	PPL	FLOPs
Full-att	-	7.55	650.19	65.37	25.97
Sparse_B	-	0.43	4756.6	57.71	1.74
Sparse_A	-	7.55	792.48	**1.00**	25.90
NSA	-	0.20	312.16	1.20	0.10
Flash-att	-	0.10	103.31	1.18	0.11
RetNet	-	0.10	226.03	1.18	0.11
Poly Basis	Three-degree	0.03	45.94	1.35	0.02
	Six-degree	0.03	70.46	1.54	0.02
	Nine-degree	0.03	90.86	1.46	0.02
Fourier Basis	Three-series	0.03	61.87	1.18	0.23
	Six-series	0.03	129.36	1.30	0.23
	Nine-series	0.03	166.21	1.31	0.23
Wavelet Basis	Harr	0.03	**37.98**	2.97	0.02
	Meaican hat	0.03	145.71	1.28	0.02
	Morlet	0.03	140.83	1.28	0.02
Gaussian Basis (μ, σ^2, n)	(1,1,1)	0.029	137.07	1.26	0.03
	(1,1,3)	0.029	140.67	1.54	0.02
	(0.5,1,1)	0.029	140.58	1.24	0.02
	(1.5,1,1)	0.029	139.58	1.27	0.02
	(1,0.5,1)	0.029	135.42	2.02	0.02
	(1,1.5,1)	0.029	140.16	1.18	0.02
Spline Basis	Linear	0.029	549.75	173.5	0.02
	Cubic	0.029	754.85	170.1	0.02
	Cubic-B	0.029	759.73	170.1	0.02

thod that simulates the NSA approach. The extra-para metric indicates the number of additional parameters introduced beyond the baseline model, while training time, PPL, and FLOPs are used to evaluate the model's adaptability across different tasks.

Our model was configured with a batch size of 32, a learning rate of 3e-4, and trained for 5 epochs, selecting the best results. The sequence length was 128, with 6 layers, hidden dimension d = 256, and 2 attention heads, using a random seed of 42. For basis function selection, we experimented with five distinct types: polynomial basis functions, Fourier basis functions, Gaussian basis functions, spline basis functions, and wavelet basis functions. Each basis function was tested with corresponding controlled experiments. The results are shown in Table 4.

5 Conclusion

We first validated the existence of an implicit value space underlying the data, which allows us to define vectors using functions, specifically, in the form of parametric equations. The advantage of this approach is that function projection calculations require significantly less computational cost compared to matrix projections, eliminating the need to update large matrix parameters for the sampling process. We acknowledge that this method demands additional computational resources during the word embedding process. However, we consider this an acceptable pretraining strategy. Finally, our experiments confirmed that function projections can reduce model complexity to some extent. We tested various types of basis functions to evaluate multiple metrics, though this represents only a small subset of possible functions. We believe this concept has significant room for improvement, and we will further explore it in future studies.

Acknowledgments. The fund of Supporting the Reform and Development of Local Universities (Disciplinary Construction) and the special research project of First-class Discipline of Inner Mongolia A. R. of China under Grant YLXKZX-ND-036.

References

1. Hassanin, Mohammed, et al. "Visual attention methods in deep learning: An in-depth survey." Information Fusion 108 (2024): 102417.
2. Guo, Meng-Hao, et al. "Attention mechanisms in computer vision: A survey." Computational visual media 8.3 (2022): 331-368.
3. Hu, Dichao. "An introductory survey on attention mechanisms in NLP problems." Intelligent Systems and Applications: Proceedings of the 2019 Intelligent Systems Conference (IntelliSys) Volume 2. Springer International Publishing, 2020.
4. Lee, John Boaz, et al. "Attention models in graphs: A survey." ACM Transactions on Knowledge Discovery from Data (TKDD) 13.6 (2019): 1-25.
5. Hu, Kai, et al. "An overview: Attention mechanisms in multi-agent reinforcement learning." Neurocomputing (2024): 128015.
6. Kalyan, Katikapalli Subramanyam. "A survey of GPT-3 family large language models including ChatGPT and GPT-4." Natural Language Processing Journal 6 (2024): 100048.
7. Yao, Yifan, et al. "A survey on large language model (llm) security and privacy: The good, the bad, and the ugly." High-Confidence Computing (2024): 100211.
8. Farina, Mirko, et al. "Sparsity in transformers: A systematic literature review." Neurocomputing (2024): 127468.
9. 13.Forrest, Stephanie. "Genetic algorithms." ACM computing surveys (CSUR) 28.1 (1996): 77-80.
10. Aggarwal, Ada, et al. "A Detailed Overview of Quantum Computing Machine Learning Techniques." 2024 International Conference on Communication, Computer Sciences and Engineering (IC3SE). IEEE, 2024.
11. Izenman, Alan Julian. "Introduction to manifold learning." Wiley Interdisciplinary Reviews: Computational Statistics 4.5 (2012): 439-446.
12. Yang, Ling, et al. "Diffusion models: A comprehensive survey of methods and applications." ACM Computing Surveys 56.4 (2023): 1-39.

13. Bozorgasl, Zavareh, and Hao Chen. "Wav-KAN: Wavelet Kolmogorov-Arnold Networks." Available at SSRN 4835325 (2024).
14. Uteuliyeva, Malika, et al. "Fourier neural networks: A comparative study." Intelligent Data Analysis 24.5 (2020): 1107-1120.
15. Biamonte, Jacob, et al. "Quantum machine learning." Nature 549.7671 (2017): 195-202.
16. Han, Chi, et al. "Word Embeddings Are Steers for Language Models." Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2024.
17. Michoski, Craig, et al. "Solving differential equations using deep neural networks." Neurocomputing 399 (2020): 193-212.
18. Cai, Shengze, et al. "Physics-informed neural networks (PINNs) for fluid mechanics: A review." Acta Mechanica Sinica 37.12 (2021): 1727-1738.
19. Child R, Gray S, Radford A, et al. Generating long sequences with sparse transformers[J]. arXiv preprint arXiv:1904.10509, 2019.
20. Qiu, Jiezhong, et al. "Blockwise Self-Attention for Long Document Understanding.", 2019.
21. Wang, Sinong et al. “Linformer: Self-Attention with Linear Complexity.” ArXiv abs/2006.04768 (2020).
22. Kitaev, Nikita, Lukasz Kaiser, and Anselm Levskaya. "Reformer: The Efficient Transformer." International Conference on Learning Representations, 2020.
23. Liu, Hao, Matei Zaharia, and Pieter Abbeel. "RingAttention with Blockwise Transformers for Near-Infinite Context." The Twelfth International Conference on Learning Representations, 2023.
24. Beltagy, Iz, Matthew E. Peters, and Arman Cohan. "Longformer: The Long-Document Transformer.", 2020.
25. Bojanowski, Sainbayar Sukhbaatar Edouard Grave Piotr, and Armand Joulin. "Adaptive Attention Span in Transformers.", 2019.
26. Han, Dongchen, et al. "Agent Attention: On the Integration of Softmax and Linear Attention." European Conference on Computer Vision. 2024.
27. Nagrani, Arsha, et al. "Attention bottlenecks for multimodal fusion." Advances in neural information processing systems 34 (2021): 14200-14213.
28. Luohe, Shi, et al. "Keep the Cost Down: A Review on Methods to Optimize LLM’s KV-Cache Consumption." First Conference on Language Modeling.
29. Liu, Akide, et al. "Minicache: Kv cache compression in depth dimension for large language models." Advances in Neural Information Processing Systems 37 (2024): 139997-140031.
30. Adnan, Muhammad, et al. "Keyformer: Kv cache reduction through key tokens selection for efficient generative inference." Proceedings of Machine Learning and Systems 6 (2024): 114-127.
31. Hooper, Coleman, et al. "Kvquant: Towards 10 million context length llm inference with kv cache quantization." Advances in Neural Information Processing Systems 37 (2024): 1270-1303.
32. Zhang, Tianyi, et al. "Kv cache is 1 bit per channel: Efficient large language model inference with coupled quantization." Advances in Neural Information Processing Systems 37 (2024): 3304-3331.
33. Hudson, Drew A., and Christopher D. Manning. "Gqa: A new dataset for real-world visual reasoning and compositional question answering." Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019.
34. Li Y, Si S, Li G, et al. Learnable fourier features for multi-dimensional spatial positional encoding[J]. Advances in Neural Information Processing Systems, 2021, 34: 15816-15829.

35. Wang, Sifan, Hanwen Wang, and Paris Perdikaris. "On the eigenvector bias of Fourier feature networks: From regression to solving multi-scale PDEs with physics-informed neural networks." Computer Methods in Applied Mechanics and Engineering 384 (2021): 113938.
36. Atchade-Adelomou, Parfait, and Kent Larson. "Fourier Series Weight in Quantum Machine Learning." (2024).
37. Yi, Kun, et al. "Frequency-domain mlps are more effective learners in time series forecasting." Advances in Neural Information Processing Systems 36 (2023): 76656-76679.
38. Karras, Tero, et al. "Alias-free generative adversarial networks." Advances in neural information processing systems 34 (2021): 852-863.
39. Finkelstein, Lev, et al. "Placing search in context: The concept revisited." Proceedings of the 10th international conference on World Wide Web. 2001.
40. Kiela, Douwe, Felix Hill, and Stephen Clark. "Specializing word embeddings for similarity or relatedness." Proceedings of the 2015 conference on empirical methods in natural language processing. 2015.
41. Bruni, Elia, et al. "Distributional semantics in technicolor." Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2012.
42. Rubenstein, Herbert, and John B. Goodenough. "Contextual correlates of synonymy." Communications of the ACM 8.10 (1965): 627-633.
43. Zhong, Yaofeng Desmond, Biswadip Dey, and Amit Chakraborty. "Symplectic ODE-Net: Learning Hamiltonian Dynamics with Control." International Conference on Learning Representations.
44. PBW-EXTENSIONS, DERIVATIONS OF SKEW. "Communications in Mathematics and Statistics."
45. Lu, Yiping, et al. "Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations." International conference on machine learning. PMLR, 2018.
46. Zhang, Xingcheng, et al. "Polynet: A pursuit of structural diversity in very deep networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.
47. Chen, Ricky TQ, et al. "Neural ordinary differential equations." Advances in neural information processing systems 31 2018.
48. Grathwohl, Will, et al. "FFJORD: Free-Form Continuous Dynamics for Scalable Reversible Generative Models." International Conference on Learning Representations. 2018.