Multi-Spot Transformer: An Efficient Transformer with Linear Complexity

Anonymous Authors[1]

Abstract

Transformers have been widely applied in text, vision, and other tasks due to their effective modeling of long range dependencies between tokens. However, their computation complexity and memory footprint are quadratic in terms of the token sequence length because they establish dense dependencies between every pair of tokens. This limits their application in tasks involving long sequences. Many works have addressed this issue by exploiting the sparsity and locality of token relationships using low-rank, kernels, routing mechanisms, or other specific patterns. However, these approaches have not fully utilized the bias in token relationships. This paper proposes a multi-spot selection mechanism to compress the corresponding KV tokens of each query token to different degrees based on their relevance, further enhancing the utilization of sparsity and achieving higher efficiency and performance. The paper also explains the role of Softmax from a filtering perspective and develops an equivalent kernel to replace Softmax, combined with the multi-spot selection mechanism to achieve a more reasonable linear attention. Additionally, the paper uses polynomial fitting with kv tokens to extend the linear fitting of the original attention, improving its expressive power. Finally, by combining the above three parts, this paper achieves linear complexity attention with a complexity of $min(O(N^2d), O(Nd^2))$, which has been validated on the ImageNet and PG19 datasets. On ImageNet, it achieves performance comparable to state-of-the-art models. A model with 120M parameters and a sequence length of 8K achieves a perplexity of 8.29 on the PG19 dataset, lower than other models with the same parameter count, and even close to the performance of Llama-7B.

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

1. Introduction

The Transformer model(Vaswani et al., 2017)(Devlin et al., 2018)(Brown et al., 2020)(Dosovitskiy et al., 2021) has garnered significant research attention due to its outstanding performance in capturing long-range dependencies. This model is built upon self-attention mechanisms and non-linear fully connected layers. The non-linear layers serve to store and memorize information, while the core self-attention mechanism calculates the similarity between different tokens and assigns weights to achieve global sequence modeling. However, the quadratic complexity of attention in modeling relationships between tokens becomes challenging for long sequences. To address this, many studies have explored techniques, such as leveraging the sparsity of sequence relationships(Child et al., 2019)(Radosavovic et al., 2020)(Kitaev et al., 2020)(Han et al., 2021) or swapping operation orders after replacing Softmax(Shen et al., 2021)(Yun et al., 2021)(Yuan & Wang, 2021), to reduce the computational complexity of attention. Despite the performance improvements offered by these strategies, they often fall short of fully revealing the intrinsic mechanisms of Transformers and exploiting the benefits of sparsity and locality in capturing dependency relationships. Therefore, this paper delves into the sparsity and locality aspects of attention modeling for dependency relationships, proposing a more flexible and efficient way to model long-range dependencies, achieving linear complexity concerning sequence length.

Firstly, several significant studies have showcased the efficacy of efficient and high-performance models achieved through regional pattern operations and sparsification. These models segment input into regions to focus on local information. Then, some models use downsampling to capture global information, followed by fixed or routing-based patterns to facilitate information exchange between regions. Examples include Swin Transformer(Liu et al., 2021) , Deformable DETR(Zhu et al., 2021), and Biformer(Zhu et al., 2023). Additionally, other models employ a global token to attend to global information, using random or fixed patterns to capture medium- to long-range information. Examples include BigBird(Zaheer et al., 2020), Longformer(Beltagy et al., 2020), and Linformer. However, the fixed or random attention strategies of these models may lack flexibility and may not effectively focus on key areas. Furthermore, exist-

ing routing and learnable attention mechanisms often fail to attend to a sufficient number of tokens. To address these limitations, we introduce a relevance selection mechanism and a learnable relevance compression mechanism to flexibly and efficiently attend to a large number of tokens, thereby achieving higher efficiency and performance.

Secondly, many papers seek to replace softmax for lower computational complexity, abstracting the output as $sim(Q,K)V$. They use separable operations like $f_1(Q)f_2(K)^T$ to achieve an equivalent of $sim(Q,K)$. As result, the output will be $f_1(Q)(f_2(K)^TV)$, which reduces the original complexity from $O(N^2d)$ to $O(Nd^2)$, as seen in Linear attention(Katharopoulos et al., 2020), Performer(Choromanski et al., 2022), and Random Feature Attention(Peng et al., 2021). Recently, Flatten Transformer(Han et al., 2023) has achieved good results by using positive features and polynomial transformations to align tokens in the embedding space with similar positive features. However, these works often lack a deep understanding of the fundamental role of Softmax, which can lead to performance degradation in some cases and make it difficult to adapt to different scenarios. Therefore, this paper provides a thorough analysis of the role of Softmax and proposes a flexible improvement to Attention by using an equivalent filtering effect of Softmax, thus ensuring performance while being adaptable to various scenarios.

Finally, the linear transformation of the final output in Transformer, which is a linear fit of V, may limit its expressive power. This paper considers introducing polynomial fitting to enhance expressiveness and improve the interpretability of the model. NTK(Jacot et al., 2018), (Wu et al., 2022), and (Dubey et al., 2022) have demonstrated that polynomials offer higher extrapolation performance and interpretability compared to traditional networks. The detailed Related work in Appendix A.1.

Building upon the existing research, this paper conducts an in-depth analysis of the original Attention mechanism by decomposing it into three parts. Subsequently, three methods are proposed: the Correlation Selection Compression mechanism (Multi-Spot), the equivalent replacement of Softmax filtering, and polynomial fitting. These methods are combined to create a Transformer model with a complexity of $min(O(N^2d), O(Nd^2))$, where N is the sequence length and d is the embedding dimension.

2. Methods

Here is the formula for Attention in Transformer, taking queries $q_i \in R^{1\times d}$, keys $K \in R^{N_k\times d}$,and values $V \in R^{N_v\times d}$ as input, where k_i and v_i are the i-th token of K and V.

$$o_i = Softmax(\frac{q_iK^T}{\sqrt{d}})V = \sum_{j=1}^{n} \frac{e^{\frac{q_ik_j}{\sqrt{d}}}}{\sum_j^n e^{\frac{q_ik_j}{\sqrt{d}}}} v_j \quad (1)$$

This paper analyzes the aforementioned Attention formula in three parts from inner to outer:

(1) using the dot product between q tokens and k tokens as a distance metric to represent their similarity, serving as the initial regression weights;

(2) the outer Softmax as a filtering operation, which scales the similarity indices, diminishing the weights of unrelated tokens close to zero, resulting in a sparser relationship that relies only on the most similar tokens for regression;

(3) the outermost layer involves a convex combination with v. This is considered as a generalized convex interpolation (detailed discussion in Appendix Theory A.2). It performs convex interpolation at corresponding positions using tokens from all positions in the entire image or text sequence. The interpolation coefficients are determined by the weights obtained from the dot product filtering. Furthermore, this convex interpolation ensures that all values are distributed within a linear hyperplane.

With the aid of figures and tables, the following sections will elaborate in detail on the theoretical aspects of the three aforementioned components. Then, modifications were made to each component to achieve a more efficient and high-performance Attention mechanism.

2.1. Multi-Spot

2.1.1. Overview

The vanilla Attention mechanism computes dot products between each pair of tokens and then uses the Softmax-filtered dot products as weights for a linear combination or interpolation of the entire sequence to determine a single token in the sequence. This approach is quadratic, inefficient, and may introduce significant noise, thereby degrading performance. Therefore, this section addresses the shortcomings of existing work by developing a more reasonable and efficient approximate Attention mechanism.

Firstly, in both image and language sequences, there is often strong locality and sparsity in token relationships. Many studies have attempted to establish local relationships using windows and then use various methods to establish relationships between regions. These methods include regional processing, sparse patterns, low-rank processing, fixed connections, or simple routing mechanisms to build information relationships between regions. However, these methods have some shortcomings. Their information capture methods are relatively fixed, or they do not select enough relevant tokens, resulting in the inability to focus on key areas or col-

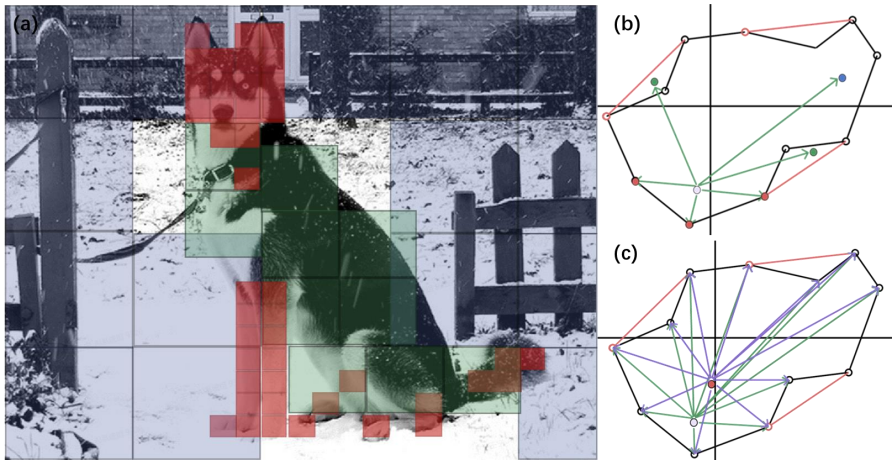

Figure 1. 1(a) Multi-scale compressed token selection mechanism: red region contains most relevant uncompressed tokens, green region contains less compressed sub-correlated tokens, and blue region contains more compressed tokens with lower correlation.1(b) Multi-scale compressed token selection mechanism in K-space: red dots represent top KV tokens most relevant to query token, green dots represent less relevant tokens compressed for semantic tokens, and blue dots represent tokens with much lower relevance compressed for semantic tokens, used for regression interpolation.1(c) Filtering effect of Softmax in K-space: compared to unfiltered red dots (linearly normalized), filtered gray dots are closer to relevant tokens. Softmax exponentially increases weights of more relevant tokens in regression interpolation and decreases weights of irrelevant tokens, suppressing the impact of irrelevant noise. The final output is a feature transformation of the output results in K-space using $W_K^{-1}W_V$, transforming them into the V-space output results.

lect information comprehensively, leading to some degree of information loss and performance degradation.

Therefore, to flexibly and adequately utilize the information in the sequence and the sparsity between tokens, this paper adopts a mechanism of relevance selection and relevance compression to construct the model.

2.1.2. Elaboration

Specifically, for each query token q, the number of kv tokens related to it is relatively sparse and flexible compared to the entire sequence. Therefore, based on the correlation between tokens, this paper uses a selection mechanism to select a sufficient number of tokens and compresses irrelevant tokens based on their correlation. Because for a query token q, the closer the similarity between kv tokens with similar correlation scores, the less impact kv tokens with smaller correlation have on q. Therefore, compressing such kv tokens with small and similar correlations by averaging or using a semantic token can ensure minimal information loss, thereby achieving efficient and high-performance approximate Attention.

For example, using the example in Figure 1(a) for illustration, when querying features of a Husky's head or legs, because the head or leg kv tokens have the greatest impact, they are selected without compression for precise information interpolation. For the body feature, because it is a holistic feature of an object and there is a certain correlation between them, but they are features with relatively small correlation and occupy a large number of pixels, using a small-scale average compression or a semantic token for interpolation can minimize information loss. Furthermore, for the surrounding environment, these features are relatively unrelated to the head or leg features, so a larger-scale average compression token can be used for rough positioning interpolation or directly filtered out.

Similarly, in the keys embedding space, illustrated in Figure 1(b), each hollow point represents a k token. Since the sum of the coefficients in the Attention combination is 1, the final convex combination of the new token will not exceed the region surrounded by k tokens. In the K-space, the red points denote features of the red region, which are the most relevant, closest to the q token, and exert the greatest impact. Thus, precise interpolation is conducted using the original values. Conversely, features with relatively small correlation, represented by the green region in Figure 1(a), are farther away from the q token. Consequently, they have a smaller impact at the same distance. Therefore, region averaging or adding a semantic token (green point) to represent the entire region is used for interpolation. Finally, for tokens that are furthest, a larger-scale compression is applied, compressing more k tokens to obtain one token (blue point), and then using the compressed tokens for interpolation.

2.1.3. Formula

Based on the analysis above, the specific interpolation formula is provided below, and the error analysis of the original formula is detailed in the Appendix A.3:

$$\sum_{m=1}^{p}\sum_{j=\sum_{\substack{n=1\\ \delta=p_m}}^{m} p_n k_n}^{p_m k_m} \frac{(\hat{p_m} e^{\frac{\sum_{t=j}^{j+p_m} d_{it}}{p_m}})}{\sum_{m=1}^{p} \hat{p_m} \sum_{j=\sum_{\substack{n=1\\ \delta=p_m}}^{m} p_n k_n}^{p_m k_m} e^{\frac{\sum_{t=j}^{j+p_m} d_{it}}{p_m}}} \sum_{s=1}^{p_m} \frac{v_s}{p_m} \tag{2}$$

where d_{it} is the dot product of $query_i$ and key_t, p is the counts of different patchsize, p_m is the patchsize of a region, k_m is the number of the p_m region, $\hat{p_m}$ is a learnable ratio.

Let

$$P_m(p_m, k_m, d, v) = \hat{p_m} \sum_{j=\sum_{\substack{n=1\\ \delta=p_m}}^{m} p_n k_n}^{p_m k_m} e^{\frac{\sum_{t=j}^{j+p_m} d_{it}}{p_m}} \sum_{s=1}^{p_m} \frac{v_s}{p_m}$$

The general choice requires $p_1 = 1$, so the first part of the equation can be expressed as $\sum_{j=1}^{k_1} e^{d_{ij}} v_j$ represents the selection of k_1 most relevant uncompressed tokens, and then $P_2(p_2, k_2, d, v)$ signifies the selection of k_2 compressed regions of size p_2 within the remaining $n-k_1$, recursively continuing with $P_m(p_m, k_m, d, v)$ representing the selection of k_m compressed regions of size p_m within the remaining $n - \sum_{i=0}^{m-1} k_i p_i$, where the patch size $p_m \geq p_{m-1}$ and the correlation $d_{.t} \geq d_{.t-1}$.

This interpolation method allows us to flexibly choose tokens of different scales and semantic sizes based on changes in relevance. By progressively selecting the most relevant uncompressed tokens and interpolating with tokens compressed to varying degrees according to their relevance, it achieves a more efficient and flexible modeling of long-distance dependencies.

2.1.4. Algorithm

Based on the above elaboration and formulas, this paper describes the algorithm used in the code in detail as follows:

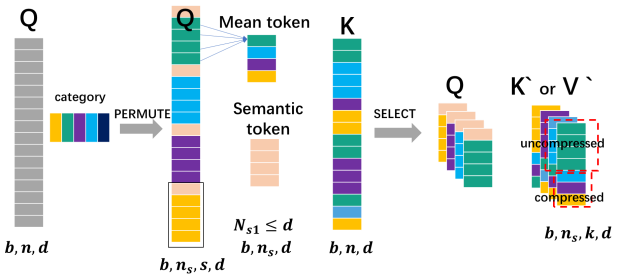

Figure 2. Schematic diagram of query Q partitioning method and KV token selection

Firstly, considering the local nature of the data, Q is divided into regions, as illustrated in Figure 2. It is partitioned into N_s regions of size s. Alternatively, when dealing with sequences that lack strong locality, a classification matrix can be used to categorize and color tokens. Subsequently, a selection mechanism (permutation) is employed to move tokens belonging to the same class to adjacent positions, thereby increasing the number of tokens that can be shared after region partitioning. The final output can use the same selection mechanism or permutation as the first time, ensuring an output sequence identical to the original.

As shown in the right half of Figure 2, representatives of Q, obtained after partitioning, are used for attention with K. The top-k most relevant uncompressed KV tokens to the query token region are selected. Subsequently, with decreasing relevance, different levels of compressed tokens are gradually introduced. These compression methods include average compression, autoencoder compression, or state space compression.The average compression entails the averaging of selected tokens, while autoencoder compression involves the fusion of information from multiple tokens using semantic tokens similar to the cls token. This approach efficiently leverages information within the sequence, as it flexibly selects multi-scale tokens that need attention for different classes of query tokens.

Through these improvements, the interpolation method proposed in our work strikes a better balance between flexibility and performance, achieving efficient modeling of long-range dependency relationships.

2.1.5. Experiment

Table 1. Model modification results with accuracy and memory usage (batchsize=400).

Model	Modify	Accuracy (%)	Memory (MB)
ViT_base_p16_384	origin	84.2	14712
	p_s36_kvtop64	73.9	7627
	p_s36_kvtop64_s16	80.7	8108
	p_s36_kvtop96_s16	81.9	9586
	p_s36_kvtop96_s4	83.1	10286
MST	onlylocal	81.4	4305
	linear_norm	73.0/81.2	4468
	3_order	79.1/81.3	5377
	polymonial	79.4/81.4	6602
	f(Q)(f(K)V)	80.1	5464

In order to verify the efficiency and effectiveness of the above theory, the following simple experiments were conducted in this paper, the results are in Table 1.

Using ViT-base-p16-384 as an example, the weights loaded in this paper achieve an accuracy of 84.2% on ImageNet. Applying the algorithm described above without fine-tuning,

the experiment first divides the Q sequence of length 576 into 16 regions of length 36, and then selects the representative kv tokens for each region that are most relevant:

(1) Selecting the top 64 most relevant tokens (topk=64) results in an accuracy of only 62.7%. After applying the denominator correction of Formula (1), the accuracy reaches 73.9%, which is higher than using the original Attention denominator correction. This indicates that 64/576=11% of kv tokens account for 73.9/84.2=88% of the performance.

(2) For topk=96, the accuracy is 78% after using the original Attention denominator correction, and reaches 80.1% after applying the denominator correction of Formula (1). 96/576=16.7% of kv tokens account for 80.1/84.2=95.2% of the performance.

(3) Further, using k=64 uncompressed tokens and the average compressed tokens of 36 regions with p=16 tokens for regression interpolation results in an accuracy of 80.9%, which is higher than the accuracy of 80.2% achieved by using only topk=100 and correcting the coefficients. Further improvement in accuracy is achieved with the use of multiple scales.

(4) Using topk=96 and p=36, the accuracy can be increased to 81.9%, which is much higher than using topk=96 alone. Further, using topk=96 and p=4, an accuracy of 83.1% is achieved, which is only a close to 1% loss compared to the original ViT accuracy.

2.1.6. Conclusion

The above results indicate that using this multi-scale approach can closely approximate the original Attention mechanism, and even may surpass it without the influence of training, as many noise tokens are filtered out. Additionally, this selection mechanism can be fully transferred to all Transformer models, allowing the interpolation of desired tokens by flexibly using tokens learned in the model. These insights contribute not only to a deeper understanding of the operational mechanisms of the model but also provide strong inspiration for future research and optimization efforts.

2.2. Softmax Nonlinear Filtering Effect

2.2.1. Elaboration

This subsection investigates the role of Softmax. Firstly, based on the theoretical formula of Softmax, it applies exponential scaling to distances represented through inner products. In contrast to linear normalization, Softmax makes weights with larger inner products even larger and weights with smaller inner products even smaller. After removing the normalization effect, the stretching ratio is $r = \frac{e^{d_2}}{e^{d_1}} : \frac{d_2}{d_1}$.

When $d_1 \geq 1$,

$$r = \frac{e^{\Delta d}}{1 + \frac{\Delta d}{d_1}} > \frac{1 + \Delta d + \frac{(\Delta d)^2}{2}}{1 + \frac{\Delta d}{d_1}} \geq \max\{1, \frac{\Delta d}{2}\}$$

where $\Delta d = d_2 - d_1$ and $d_2 \geq d_1$. As $d_1 \leq -1$, Linear normalization uses negative values for interpolation, while Softmax shrinks its weight exponentially to suppress its effect. The $-1 < d_1 < 1$ interval is confusing, but does not play a key role in interpolation due to its small value.

This way enables a higher emphasis on interpolation using nearby points that share more similarity, resulting in improved effectiveness, as illustrated in Figure 1 (c). The purple line, representing the linear approach, evenly interpolates all points to generate the red point, while the green line of the Softmax approach tends to interpolate based on more similar points, producing the gray point.

Essentially, Softmax acts as a filtering mechanism by excluding or suppressing other unrelated points, selecting the most relevant points for interpolation. Simultaneously, it achieves greater sparsity, as the exponential reduction leads to many smaller values. This sparsity can further enhance performance, runtime efficiency, and reduce memory usage. However, Softmax also faces certain challenges, particularly when dealing with pseudo-correlated points and outliers. In the presence of highly correlated single-point noise, the exponential nature of Softmax may result in significant errors.

2.2.2. Experiment

To validate the perspective of interpolating based on more similar points, we conducted a series of experiments using the Vision Transformer (ViT) and our self-trained Multi-Spot Transformer (MST) model. The MST-T-S4-C1 selects the most relevant uncompression tokens for interpolation.

section 2.1 indicates that the most relevant 96 tokens contain the vast majority of information from each region, demonstrating significant sparsity and effectively filtering out the influence of irrelevant elements. Furthermore, to eliminate the impact of Softmax during training, we retrained the MST-T-S4-C1 model, which selectively uses the most relevant 64 tokens for regression interpolation. Its final performance is 81.4, achieving 97% of the original performance with only 11% of the tokens, and increasing the parameter quantity to ViT-base yields even better results.

To further investigate the role of Softmax, during inference, we replaced the $Softmax(dots)$ in the MST-T-S4-C1 model with as follow, the results also in Table 1:

(1) $dots/sum(abs(dots))$, resulting in an accuracy drop to 73.0, which is 90% of the original performance.

(2) $dots^n/sum(abs(dots^n))$, the accuracy gradually approached 81.4 with increasing values of n.

(3) Additionally, replacing with $dots^3/sum(abs(dots^3))$, and retraining for 10 more epochs, restored the performance to 81.3. This indicates that Softmax is not indispensable, as its primary role is to filter and select a series of most relevant points for interpolation.

(4) After replacing Softmax with separable functions and retraining for an additional 10 epochs, the performance was restored to 80.9.The ability to linearly replace Softmax is attributed to the earlier use of the top-k selection method, which to some extent serves as a filtering effect, mitigating the impact of Softmax. It also underscores the effectiveness of approximating element interpolation.

(5) Moreover, to nullify the beneficial impact of Softmax during the training of the MST-T-S4-C1 model, we replaced Softmax with $dots/sum(abs(dots))$ and retrained it from scratch, resulting in a performance of 81.2, very close to the performance using Softmax.

2.2.3. Conclusion

So from the above, we further summarize a separable kernel function based on its filtering effect and it is one of the components for the subsequent implementation of linear complexity model.

Theory 2-2-1: $f(pa(q_{ij}))f(pa(k_{ij}))$ replaces $q_{ij}k_{ij}$ can achieve a similar filtering effect, where $f(x)$ satisfies the condition that its derivative $f'(x) \geq cx^{\frac{1}{n}}$, (when $x > 0$),where $c > 0$, $n > 0$ and pa is positive activation. As n gets larger the filtering effect gets sharper and can be set as a learnable parameter.

Examples of such functions include simple functions like $x^{1+\frac{1}{n}}, e^x$, or compound functions that meet the derivative condition. Here, pa represents positive activation, an activation function that ensures features are activated as positive values and maintains positive monotonicity. The derivative of pa is required to be greater than or equal to c, and examples include ReLU, Softplus, Swish, or functions satisfying the above two conditions. In this case, the above formula can achieve specific scaling, thus realizing a filtering effect similar to Softmax. Detailed proofs are available in the appendix.

In summary, both theoretical analysis and experimental results indicate that using the replacement of more similar tokens for interpolation, along with linear or simple high-power functions, yields effects comparable to Softmax approximation. Therefore, it is reasonable to consider employing an equivalent decomposition of QK with a selection mechanism to achieve linear complexity with respect to sequence length, thereby improving performance.

2.3. polynomial regression

$$o_i = \sum_{j=1}^{n} A_{ij} v_j, o_i^2 = \sum_{j=1}^{n} A_{ij} v_j^2,$$

Based on the output formula above, it can be seen that out is obtained by a linear convex combination of v, which is equivalent to fitting a linear hyperplane to a manifold that can complete the task's true data distribution. If both o and v are replaced with quadratics, and v is normalized, then according to $\sum_{j=1}^{n} A_{ij} = 1$, the output lies on a hypersphere. Furthermore, a neural network can be approximated as piecewise polynomial fitting to a true function that can complete the task (for detailed theoretical explanation, see the Appendix A.6). According to the above formula, Attention internally uses linear fitting. Therefore, according to polynomial theory, we can study the use of low-order polynomials of v to increase the order and number of interactions of piecewise polynomials, thereby enhancing its expressive power.

Experimental verification is as follows: when using interpolation with two scales of elements, the accuracy of regression using the original v is 81.6. The accuracy of regression using $av^3 + bv^2 + v$ reaches 81.9, showing a slight improvement. Here, v^p is not just an element-wise power operation, but a high-order interaction arranged according to complexity requirements, involving $v_{ij}v_{mn}$.

Furthermore, after training MST-T, the accuracy obtained is 82.0%. Further fine-tuning using $av^3 + bv^2 + v$ to replace v, with only a, b, and the parameter weights for obtaining qkv being updated, can increase the accuracy to 82.1% after 10 epochs.

However, due to the excessive memory consumption introduced by $av^3 + bv^2 + v$, and its tendency to overfit, this model rarely incorporates it during training. It is mostly used for performance improvement after training is completed.

3. Linear complexity model

Based on the analyses in the three preceding sections, we have devised a highly flexible model that achieves a favorable balance between performance and computational complexity. The computational complexity of the model is given by $min(O(CN^2d), O(CNd^2))$. C is a small constant typically not exceeding 4.

3.1. Modules

The algorithmic description of the model is outlined below:

(1) For the division of Q regions, first classify Q, such as using linear complexity KNN or classification matrix, and

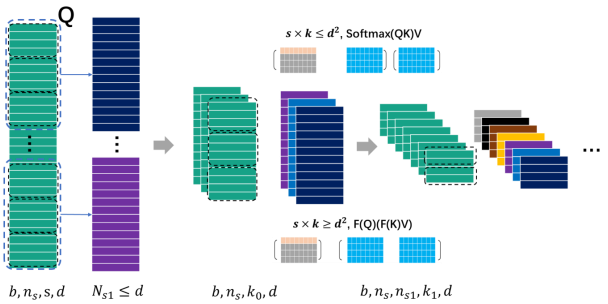

Figure 3. Linear Module Work Process. Utilizing multi-level semantic tokens or compressed tokens to first select the most relevant kv tokens in a large region, then progressively select tokens in smaller regions. Finally, if the number of kv tokens shared by each q region is greater than dim, a kernel function is used to reduce complexity; otherwise, Softmax is used.

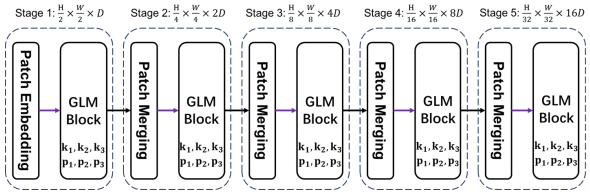

Figure 4. Model Architecture Schematic

replace similar tokens with nearby ones. Then, Q is divided into multiple levels, as shown in the leftmost part of Figure 3, where the Q token sequence is divided into three levels: the highest level l_2, which contains the most tokens, approximately $\frac{N}{Cdim}$; the next highest level l_1, which contains a certain number of tokens, approximately $\frac{1}{Cdim}$ of the previous level; and the lowest level l_0, which contains individual tokens.

(2) Use preprocessed Q to select KV tokens related to each region. First, select a large number of tokens related to this region using the representative of the highest level l_n. Then, in the next lower level, select tokens related to their own region from the KV tokens selected in the previous level, until l_1. Because similar q tokens share a large number of the same kv tokens, there are also a large number of q tokens in l_1, which is also one of reason why the final complexity is linear.

(3) After the selection, determine if the number of tokens in l_1 is greater than dim. If so, use the separate kernel method for the final Attention; otherwise, use Softmax for Attention.

The above method achieves linear complexity of the model, with a complexity of $min(O(N^2d), O(Nd^2))$. For a detailed complexity analysis, see the Appendix A.7. Another reason for not directly using the separate kernel method for Attention to achieve linear complexity is that for sequences that are very long, there is a large amount of noise. Directly using this method for filtering may not yield ideal results. Therefore, using a selection mechanism for pre-filtering is a very good approach.

Ultimately, when using v for regression interpolation, one can consider using a low-order polynomial of v to enhance the final fitting expressiveness. For example, employing an expression like $\alpha v^3 + \beta v^2 + v$ as described in section 2.3.

3.2. model architecture

The overall architecture of the model is divided into five stages. In the first stage (Depending on the actual task and complexity effects can choose not to use the), a resolution of H/2*W/2 is utilized, and during the selection process, Q is partitioned twice. In each region, the most relevant k1 tokens, k2 sub-relevant tokens with compression p2, and tokens compressed more globally are selected. The attention operation involves kernel method. In the second and third stages, a single Q region partition is employed, selecting three types of tokens with different compression ratios. Softmax is used as the filtering mechanism. In the fourth and fifth stages, the original attention mechanism is directly applied. Additionally, each stage can be configured to use different modes. For example, the fourth stage can utilize the selection mechanism, leading to improved performance and reduced memory consumption, albeit at the cost of potentially lower operational efficiency.

3.3. Experiment

3.3.1. ImageNet

The designed model in this paper underwent experiments on ImageNet, training for 300 epochs with the AdamW optimizer and a weight decay of 0.05. To prevent overfitting, data augmentation techniques such as mixup were applied, and the configuration matched that of Swin Transformer. The results are presented in Table 2.

(1) First, the accuracy of MST-T-S4-C3 without stage 1 and polynomial fitting is 82.0, which is 2.6 higher than VVT-T(Sun et al., 2023), is 0.7 higher than Swin-T and 0.6 higher than Biformer-T(Zhu et al., 2023). The utilization rate of GPU memory and the execution speed are both comparable to or better than Biformer.

(2) MST-T-S5-12K, an extension of MST-T-S4-C3 with an additional layer of stage 1 block and two rounds of region partitioning, exhibits a minor increase in memory overhead without a corresponding improvement in accuracy. Possible explanations include the insufficient ImageNet dataset, unable to support the increased precision interpolation brought by higher resolution, or the classification task itself may not

Table 2. Classification accuracies and model specifications for various models.T:tiny,S:small,S4/5:4/5 stages,C1/2/3:select 1/2/3 types of compressed tokens,P3:3rd order polynomial.

MODEL	FLOPS (G)	PARAMS (M)	TOP-1 ACC. (%)
RESNET-18	1.8	11.7	69.8
REGNETY-1.6G	1.6	11.2	78
PVTV2-B1	2.1	13.1	78.7
VVT-T	3.0	12.9	79.4
SHUNTED-T	2.1	11.5	79.8
QUADTREE-B-B1	2.3	13.6	80
BIFORMER-T	2.2	13.1	81.4
MST-T-S4-C1	2.0	13.1	81.4
MST-T-S4-C2	2.2	13.1	81.6
MST-T-S4-C2-P3	2.3	13.1	81.9
MST-T-S4-C3	2.5	13.1	82.0
MST-T-S5-12K	2.7	13.4	82.0
SWIN-T	4.5	29	81.3
CSWIN-T	4.5	23	82.7
DAT-T	4.6	29	82
CROSSFORMER-S	5.3	31	82.5
REGIONVIT-S	5.3	31	82.6
QUADTREE-B-B2	4.5	24	82.7
VVT-S	5.6	25.5	82.7
MAXVIT-T	5.6	31	83.6
SCALABLEVIT-S	4.2	32	83.1
UNIFORMER-S*	4.2	24	83.4
BIFORMER-S	4.5	26	83.8
MST-S-S4	4.3	26	83.7

Table 3. The experiment on PG19 datasets. Landmark Attention(Mohtashami & Jaggi, 2023), llama(Touvron et al., 2023)

MODEL	PARAMS	PERPLEXITY
LANDMARK-4K	128M	14.72
LLAMA-7B-4K	7B	7.77
MST-NLP-8K	121M	8.29

require high-resolution information.

(3) Subsequently, for the MST-S-S4 model with 26 million parameters, the accuracy reaches 83.7. This performance is comparable to state-of-the-art visual models like BiFormer-S.

The strength of this model lies in its flexibility to tailor selection parameters and compression levels, making it adaptable to various tasks and scenarios. After training with a large selection count, the model can be deployed with a smaller selection count for rapid inference with a significant reduction in GPU memory requirements.

3.3.2. PG19

The model trained in this paper on PG19 is configured as follows: with a total of 120M parameters, training sequence length of 8K, 20 layers, embedding dimension of 512, and 8 heads. Among the 8 heads, 4 heads use local-Attention with a window size of 256, and the other 4 heads use the method proposed in this paper. Specifically, 256 most relevant tokens are selected, along with 128 compressed tokens with $p = 4$ and 64 globally compressed tokens. Owing to resource issues, the model is trained for 20 epochs, only approximately 70,000 steps, using AdamW optimizer. The tokenizer used is the GPT2-tokenizer, same as landmark Attention, and perplexity evaluation is performed using the Hugging Face's evaluation method.

As shown in Table 3, our 120M model achieves a perplexity of 8.29 on PG19, surpassing the landmark model and approaching llama-7B.

3.3.3. ABLATION STUDY ON IMAGENET

Table 4. Ablation Study

MODEL	TOP-1 ACC. (%)
MST-T-S4-C1	81.4
+MULTI-SPOT	82.0
+POLY	82.1
+KERNEL	82.0

The ablation experiments demonstrate that using Multi-Spot can improve performance without reducing efficiency. Additionally, polynomial fitting can also improve performance, but may introduce unnecessary memory usage, hence it is recommended for use during fine-tuning. Finally, using separable kernel functions can maintain performance, but the specific efficiency improvement needs to be designed according to the specific task, even though it is a linear complexity method.

4. Conclusion

This paper begins by dissecting the Transformer formula, dividing it into three key components: for the inner product of q and k, we employ a multi-scale selection mechanism to replace the original inefficient calculation of similarity across the entire sequence; concerning the Softmax operation, this paper analyzes its role as a filtering mechanism, enabling a more essential and flexible replacement for improved performance or reduced complexity; finally, the paper utilizes a polynomial interpolation of v to enhance the model's expressive power, asserting that this approach is generalizable and can enhance any linear fitting method. In conclusion, based on the analysis of these three components, this paper introduces a highly flexible model that achieves linear complexity through multiple divisions in the selection of KV tokens and the separable kernel function under certain conditions, leveraging multi-scale tokens to enhance performance.

References

Beltagy, I., Peters, M. E., and Cohan, A. Longformer: The long-document transformer, 2020.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners, 2020.

Child, R., Gray, S., Radford, A., and Sutskever, I. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

Choromanski, K., Likhosherstov, V., Dohan, D., Song, X., Gane, A., Sarlos, T., Hawkins, P., Davis, J., Mohiuddin, A., Kaiser, L., Belanger, D., Colwell, L., and Weller, A. Rethinking attention with performers, 2022.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, 2018.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.

Dubey, A., Radenovic, F., and Mahajan, D. Scalable interpretability via polynomials, 2022.

Fan, F.-L., Li, M., Wang, F., Lai, R., and Wang, G. On expressivity and trainability of quadratic networks. *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–15, 2023. doi: 10.1109/TNNLS.2023.3331380.

Fedus, W., Zoph, B., and Shazeer, N. Switch transformers: scaling to trillion parameter models with simple and efficient sparsity. *J. Mach. Learn. Res.*, 23(1), jan 2022. ISSN 1532-4435.

Han, D., Pan, X., Han, Y., Song, S., and Huang, G. Flatten transformer: Vision transformer using focused linear attention. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5961–5971, October 2023.

Han, K. et al. Cswin transformer: A general vision transformer backbone with cross-shaped windows. *arXiv preprint arXiv:2107.00652*, 2021.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/5a4be1fa34e62bb8a6ec6b91d2462f5a-Paper.pdf.

Katharopoulos, A., Vyas, A., Pappas, N., and Fleuret, F. Transformers are RNNs: Fast autoregressive transformers with linear attention. In III, H. D. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5156–5165. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/katharopoulos20a.html.

Kitaev, N., Kaiser, Å., and Levskaya, A. Reformer: The efficient transformer. In *Proceedings of the International Conference on Learning Representations*, 2020.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10012–10022, October 2021.

Mohtashami, A. and Jaggi, M. Landmark attention: Random-access infinite context length for transformers, 2023.

Peng, H., Pappas, N., Yogatama, D., Schwartz, R., Smith, N. A., and Kong, L. Random feature attention, 2021.

Radosavovic, I., Kosaraju, R. P., Girshick, R., He, K., and Dollár, P. Data-efficient image transformer. In *Advances in Neural Information Processing Systems*, pp. 14167–14179, 2020.

Shen, Z., Zhou, Y., Bai, R., and Li, J. Kernelized self-attention for generating long sequences. In *arXiv preprint arXiv:2106.13254*, 2021.

Sun, W., Qin, Z., Deng, H., Wang, J., Zhang, Y., Zhang, K., Barnes, N., Birchfield, S., Kong, L., and Zhong, Y. Vicinity vision transformer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(10):12635–12649, 2023. doi: 10.1109/TPAMI.2023.3285569.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Ferrer, C. C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn,

A., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I., Korenev, A., Koura, P. S., Lachaux, M.-A., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E. M., Subramanian, R., Tan, X. E., Tang, B., Taylor, R., Williams, A., Kuan, J. X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., and Scialom, T. Llama 2: Open foundation and fine-tuned chat models, 2023.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Å., and Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*, pp. 5998–6008, 2017.

Wang, S., Li, B. Z., Khabsa, M., Fang, H., and Ma, H. Linformer: Self-attention with linear complexity, 2020.

Wu, Y., Zhu, Z., Liu, F., Chrysos, G., and Cevher, V. Extrapolation and spectral bias of neural nets with hadamard product: a polynomial net study. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A. (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 26980–26993. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/acb3565a58dea4c39c84af35d4225d97-Paper-Conference.pdf.

Yuan, L. and Wang, Y. Tokens-to-token vit: Training vision transformers from scratch on imagenet. In *arXiv preprint arXiv:2101.11986*, 2021.

Yun, H., Han, W., and Kim, J. Global context transformer: The power of the word embeddings. In *arXiv preprint arXiv:2106.09681*, 2021.

Zaheer, M., Guruganesh, G., Dubey, K. A., Ainslie, J., Alberti, C., Ontanon, S., Pham, P., Ravula, A., Wang, Q., Yang, L., and Ahmed, A. Big bird: Transformers for longer sequences. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 17283–17297. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/c8512d142a2d849725f31a9a7a361ab9-Paper.pdf.

Zhu, L., Wang, X., Ke, Z., Zhang, W., and Lau, R. W. Biformer: Vision transformer with bi-level routing attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10323–10333, June 2023.

Zhu, X., Su, W., Lu, L., Li, B., Wang, X., and Dai, J. Deformable detr: Deformable transformers for end-to-end object detection, 2021.

A. appendix

A.1. Related Work

Work related to section 2.1: For instance, Swin Transformer(Liu et al., 2021) first performs self-attention within a local region, exchanges information between different regions using shift windows, and gradually merges regions to expand the receptive field. Deformable DETR(Zhu et al., 2021) utilizes reference points, computes a series of offsets about these points, and obtains corresponding sampling points as K tokens for interaction with Q. Biformer(Zhu et al., 2023) partitions Q into specific regions, selects several KV regions related to each Q region using region-averaged tokens, and performs self-attention with the partitioned Q regions and their respective selected KV tokens. This method achieves a complexity of $N^{4/3}$. BigBird(Zaheer et al., 2020) and Longformer(Beltagy et al., 2020) uses window attention to capture local characteristics, global tokens for global information. While BigBird randomly selects r tokens to mix information from other parts of the sequence, Longformer uses Dilated Sliding Window to capture information at moderate distances. Switch Transformer(Fedus et al., 2022) employs sparse routing to achieve an efficient expert mixing model. However, these models with fixed or random attention strategies for specific regions lack sufficient flexibility and may not efficiently capture crucial information, resulting in potential performance losses. On the same computational and storage resources, introducing a selection mechanism offers greater flexibility to combine relevant regions, enhancing the model's perception of various local or global information.

Work related to section 2.2: Many papers seek to replace softmax for lower computational complexity, abstracting the output as sim(Q,K)/C V. They use separable operations like $f_1(Q)f_2(K)^T$ to achieve an equivalent of sim(Q,K), enabling the exchange of orders for linear complexity, i.e., $f1(Q)(f2(K)^T V)$, which reduces the original complexity from $O(N^2d)$ to $O(Nd^2)$. Linear attention(Katharopoulos et al., 2020) uses separable kernel functions to replace softmax, swapping orders for linear complexity. Random Feature Attention(Peng et al., 2021) and Performer(Choromanski et al., 2022) utilizes different random feature method to approximate the Gaussian kernel function and softmax. Linformer(Wang et al., 2020) modifies the KV token matrix with low-rank characteristics and applies two softmax layers to achieve linearity. Flatten Transformer(Han et al., 2023) employs positive features and polynomial transformations to pull tokens in similar directions in the embedding with similar positive features. However, these papers lack a deep understanding of softmax's essential effect, which may result in performance losses in certain situations. Therefore, this paper conducts an in-depth analysis of the role of Softmax and utilizes its filtering effect as an equivalent replacement to flexibly improve Attention operations, ensuring adaptability to various scenarios while maintaining performance.

Work related to section 2.3: Finally, as the ultimate output in Transformers involves a linear fit for V, this linear operation further limits its expressive capacity. Leveraging the theories of polynomial neural networks, this paper introduces polynomial fitting to enhance expressiveness and offer a more interpretable improvement to the model. Since all functions can be approximated by piecewise polynomial functions, many studies have investigated artificial neural networks (ANNs) from the perspectives of polynomials and kernel functions. (Jacot et al., 2018) establishes a connection between ANNs and kernel methods, introducing a new kernel function called the Neural Tangent Kernel to describe the evolution of the ANN training process. Furthermore, (Wu et al., 2022) derives finite-width Neural Tangent Kernel formulas for neural networks with Hadamard products (NNs-Hp) and explains the advantages of polynomial neural networks (PNNs) over standard neural networks in extrapolation and spectral deviation. (Fan et al., 2023) uses spline interpolation, and the Scalable Polynomial Additive Model (SPAM)(Dubey et al., 2022) models high-order feature interactions to achieve an interpretable high-performance system.

A.2. approximate interpolation

A.2.1. Discussion 1

The theoretical formulae are as follows:

$$f(x,y) = \sum_{i=0}^{h} \sum_{j=0}^{w} W_{ij} \cdot f(x+i, y+j)$$

$$o(x,y) = \sum_{i=0}^{h} \sum_{j=0}^{w} g(d_{ij}) \cdot v(x+i, y+j)$$

In the above equations, $h = 2$ corresponds to bilinear interpolation, and $h = w = 3$ corresponds to cubic spline interpolation. When interpolating a point using the entire image, comparing these interpolation formulas with the Attention formula reveals that their fundamental principles are similar. Both methods involve interpolating with "points in close proximity" by combining vectors with similar features. Consequently, the interpolated results emphasize these shared features, making them more beneficial for task completion.

However, according to the definition of interpolation, it involves fitting a curve through known points. In practice, Attention does not require the values at the original positions to be known, and the resolution remains unchanged, maintaining the size of the entire image. The positions used in the above formulas are intrinsic two-dimensional position vectors of the sequence. However, in models, Attention often incorporates positional encodings, whether relative, absolute, or from CNN, into feature vectors as positional information. Therefore, the actual interpolation positions should align with the encoded vectors used above, representing positions in feature space rather than the true physical sequence of the image.

In feature space, it is akin to forming a linear hyperplane by convexly combining the vertices of the v matrix to fit a manifold. This approach aligns more closely with the definition of regression fitting. Thus, this paper considers it as an approximate "interpolation" in the direction of the sequence entity, essentially a regression fitting in feature space.

The approximate definition of "interpolation": The definition formula for interpolation, as shown in the equation $f(x, y) = \sum_{i=0}^{h} \sum_{j=0}^{w} W_{ij} \cdot f(x + i, y + j)$, depicts coefficients that increase as the distance determined by a certain metric decreases. Ultimately, this achieves the combination of vectors with similar features in close proximity, highlighting relevant characteristics. Note: It is not mandatory for the curve to pass through known points.

A.2.2. DISCUSSION 2

Self-Attention is an extension of the interpolation formula on Riemannian manifolds given by $Exp_{(P,\lambda)}(T) = P^{1/2} \exp(\lambda P^{-1/2} T P^{-1/2}) P^{1/2}$, where T is the tangent vector matrix, a symmetric matrix decomposable into ww^T, and P is the metric matrix corresponding to a point on the manifold. Extending this to QKV, ww^T is generalized to $w_Q w_K^T$, and $P^{1/2}$ and $P^{-1/2}$ are the X matrices in Attention. Thus, the interpolation formula becomes $Exp_{(P,\lambda)}(T) = \exp(X w_Q w_K^T X) X^{-1}$, completing the interpolation and transforming it to the V space.

A.3. Error analysis

The derivation below simplifies $d_{ij}/\sqrt{d}$ to d_{ij} without loss of generality.

Firstly, the original Attention formula is divided into p parts:

$$\sum_{m=1}^{p} \sum_{j=\sum_{n=1}^{m} p_n k_n}^{p_m k_m} \frac{e^{d_{ij}} v_j}{\sum_j^n e^{d_{ij}}}$$

The subsequent derivation takes the first part of the error, denoted with a simple subscript, as an example. The process is general and applicable to the derivation of errors for any i part. Additionally, the following derivation simplifies $d_{ij}/\sqrt{d}$ to d_{ij}.

The formula for the first part is

$$p_1 \sum_{j=0,\delta=p_1}^{p_1 k_1} e^{\frac{\left(\sum_{t=j}^{j+p_1} d_{it}\right)}{p_1}} \sum_{t=1}^{p_1} \frac{v_t}{p_1}$$

and

$$\frac{\sum_{j=1}^{p_1 k_1} e^{d_{ij}} v_j}{\sum_j^n e^{d_{ij}}}$$

Their errors are as follows:

$$\varepsilon_1 = \frac{\sum_{j=1}^{p_1 k_1} e^{d_{ij}} v_j}{\sum_j^n e^{d_{ij}}} - \frac{p_1 \sum_{j=0,\delta=p_1}^{p_1 k_1} e^{\frac{\left(\sum_{t=j}^{j+p_1} d_{it}\right)}{p_1}} \sum_{t=1}^{p_1} \frac{v_t}{p_1}}{p_1 \sum_{j=0,\delta=p_1}^{p_1 k_1} e^{\frac{\left(\sum_{t=j}^{j+p_1} d_{it}\right)}{p_1}} + \sum_{m=2}^{p} p_m \sum_{\substack{j=\sum_{n=1}^{m} p_n k_n \\ \delta=p_m}}^{p_m k_m} e^{\frac{\sum_{t=j}^{j+p_m} d_{it}}{p_m}}}$$

Generalizing the denominator of both parts yields:

$$\frac{(p_1 \sum_{j=0,\delta=p_1}^{p_1 k_1} e^{\frac{\left(\sum_{t=j}^{j+p_1} d_{it}\right)}{p_1}} \sum_{t=1}^{p_1} \frac{v_t}{p_1})(\sum_j^n e^{d_{ij}}) - (p_1 \sum_{j=0,\delta=p_1}^{p_1 k_1} e^{\frac{\left(\sum_{t=j}^{j+p_1} d_{it}\right)}{p_1}} + \sum_{m=2}^{p} p_m \sum_{\substack{j=\sum_{n=1}^{m} p_n k_n \\ \delta=p_m}}^{p_m k_m} e^{\frac{\sum_{t=j}^{j+p_m} d_{it}}{p_m}}) \sum_{j=1}^{p_1 k_1} e^{d_{ij}} v_j}{C_1 C_2}$$

First-order Taylor expansion of the numerator part of the formula:

$$p_1((k_1 + \sum_{j=0,\delta=p_1}^{p_1 k_1} \frac{\left(\sum_{t=j}^{j+p_1} d_{it}\right)}{p_1}) \sum_{t=1}^{p_1} \frac{v_t}{p_1}))(n + \sum_j^n d_{ij})$$
$$-(p_1(k_1 + \sum_{j=0,\delta=p_1}^{p_1 k_1} \frac{\left(\sum_{t=j}^{j+p_1} d_{it}\right)}{p_1}) + \sum_{m=2}^{p} p_m(k_m + \sum_{\substack{j=\sum_{n=1}^{m} p_n k_n \\ \delta=p_m}}^{p_m k_m} \frac{\sum_{t=j}^{j+p_m} d_{it}}{p_m}))((p_1 k_1 + \sum_{j=1}^{p_1 k_1} d_{ij})v_j)$$

Simplify to get:

$$p_1((k_1 + \sum_{j=0}^{p_1 k_1} \frac{d_{ij}}{p_1}) \sum_{t=1}^{p_1} \frac{v_t}{p_1}))(n + \sum_j^n d_{ij})$$
$$-(p_1(k_1 + \sum_{j=0,\delta=p_1}^{p_1 k_1} \frac{\left(\sum_{t=j}^{j+p_1} d_{it}\right)}{p_1}) + \sum_{m=2}^{p} p_m(k_m + \sum_{\substack{j=\sum_{n=1}^{m} p_n k_n \\ \delta=p_m}}^{p_m k_m} \frac{\sum_{t=j}^{j+p_m} d_{it}}{p_m}))((p_1 k_1 + \sum_{j=1}^{p_1 k_1} d_{ij})v_j)$$

Further simplification:

$$((p_1 k_1 + \sum_{j=0}^{p_1 k_1} d_{ij}) \sum_{t=1}^{p_1} \frac{v_t}{p_1}))(n + \sum_j^n d_{ij}) - ((p_1 k_1 + \sum_{j=0}^{p_1 k_1} d_{ij}) + \sum_{m=2}^{p} p_m(k_m + \sum_{j=\sum_{n=1}^{m} p_n k_n}^{p_m k_m} d_{ij}))((p_1 k_1 + \sum_{j=1}^{p_1 k_1} d_{ij})v_j)$$

$$= ((p_1 k_1 + \sum_{j=0}^{p_1 k_1} d_{ij}) \sum_{t=1}^{p_1} \frac{v_t}{p_1}))(n + \sum_j^n d_{ij}) - (n + \sum_j^n d_{ij})((p_1 k_1 + \sum_{j=1}^{p_1 k_1} d_{ij})v_j)$$

Finalize:

$$((\sum_{j=1}^{p_1 k_1} d_{ij}) \sum_{t=1}^{p_1} \frac{v_t}{p_1}) - \sum_{j=1}^{p_1 k_1} d_{ij} v_j)(n + \sum_j^n d_{ij})$$

This derivation process doesn't rely on the specific properties of the first part, and the formula above can be generalized to the l-th part:

$$(\sum_{j=1}^{p_l k_l} d_{ij}(\sum_{t=1}^{p_l} \frac{v_t}{p_l} - v_j))(n + \sum_{j}^{n} d_{ij})$$

When $p_1 = 1$, indicating the selection of the most relevant uncompressed tokens for interpolation, this part has no 0th and 1st order errors. For other dimensions p, there is a 1st order error due to compression, but since the numerator is significantly large, the loss caused by this compression is minimal. Furthermore, for unrelated tokens, this compression can further reduce noise, potentially improving performance.

A.4. Multi-spot Discussion

Furthermore, based on experiments and Transformer formulations, we observe that this interpolation method achieves a linear hyperplane fitting of the underlying learned manifold data distribution, interpolating relevant data points to positions with closely represented distances in the inner product space. In this process, the role of the feedforward neural network (FFN) is evident. It acts as a spatial transformer, preserving the data distribution on the continuously fitted transformed manifold. Therefore, in this context, the FFN plays a crucial role in storing information.

In contrast, the Attention mechanism operates flexibly with data points. By selecting relevant tokens, Attention utilizes the relevant points to interpolate new data points and expected outputs with similar features in this space. This approach also provides a good explanation for the process of context learning. By using only the relevant new input data points, based on the established manifold distribution and modeled relationships, Attention selects relevant tokens in the designed sequence to interpolate words and sentences related to the expected context. This is why designing prompts with more similar points offers a more accessible way to interpolate and regress the expected context. With more related points, the interpolation can be more precise in regressing the desired tokens. Furthermore, within the model, it is possible to randomly generate KV tokens that are close to the selected tokens, thus using more similar elements for interpolation to improve performance.

It can be further inferred that the establishment of more precise distributions and relationships requires a larger amount of training data. Consequently, in this study, experiments with the CIFAR dataset revealed that, compared to using patch=2, employing coarser compressed tokens with patch=4 for interpolation is more conducive to small datasets. Additionally, based on the relationships established through this distribution process, dataset augmentation can be achieved. For labeled data, one approach to expanding the dataset is by employing cross-interpolation using two images with similar semantics or labels.

These experiments have demonstrated the effectiveness of the interpolation method and presented a range of possibilities for enhancing flexibility and performance. These insights contribute not only to a deeper understanding of the operational mechanisms of the model but also provide strong inspiration for future research and optimization efforts.

A.5. Filtering Equivalence

Firstly, each token is composed of features from various columns, where a larger value indicates a more prominent and pronounced presence of that feature, while a smaller value signifies the absence of the feature. Of course, the numerical values themselves might have inherent meanings, displaying different effects based on their positive or negative magnitudes. When modeling similarity or distance through inner products, it fundamentally involves comparing features. If both tokens share certain features and lack others, it will positively influence the inner product. However, when comparing features, one token may have positive values and the other negative, which could negatively impact the final inner product. In Flatten Transformer(Han et al., 2023), using ReLU selectively with positive values filters out the need to check whether the original dot products are greater than zero. This allows for directly stretching $x_1 x_2$ based on the use of $f(x_1)f(x_2)$, achieving a filtering effect similar to softmax. However, relying solely on positive features may lead to misjudgments of similarity. For instance, if the original dot products are small but using only positive values results in a large inner product, as in the case of $[2, -2, 3, -5, 5]$ and $[3, 2, -1, 5, 6]$. Nevertheless, from another perspective, ReLU can be viewed as a feature activation, akin to learning to activate specific features for similarity judgments.

This paper currently focuses on finding a function for the QK operation and filtering effect that is equivalent to Softmax, under the assumption of feature activation. Initially, an activation function is applied to activate the features of input Q and V. If the features activated by the activation function do not guarantee positivity, further processing is performed to ensure that the activated features are represented in a positive manner. Subsequently, these features are used for modeling similarity

through inner product operations. Representing features as positive values facilitates the identification of simple functions to achieve a non-linear filtering effect similar to Softmax, akin to the approach used in Flatten Transformer, which is extended in this paper.

Elaborate Theory 2-2-1:

The desired function is as follows:

Given the symmetry of the QK inner product, applying the same function to both Q and K is a favorable approach. Similarly, using similar functions with identical properties on Q and K individually is also effective. However, this paper does not delve into further discussion on this topic, leaving it for exploration in another ongoing work.

This paper models the filtering properties similar to Softmax as a function that scales higher values with increasing similarity between Q and K compared to linear normalization. The degree of scaling is quantified by a power function in terms of x.

The necessary condition for achieving Softmax-like filtering effects is that the ratio of distances, $f(Q)f(K) - f(Q_0)f(K_0)$, to $QK - Q_0K_0$, should increase with the growth of Q and K. Assuming $Q_0K_0 = 0 = f(Q_0)f(K_0)$, this requires the ratio $F(Q, K) = \frac{f(Q)f(K)}{QK}$ to be a positive increasing function. Taking the partial derivative with respect to Q (considering symmetry with respect to Q and K, focusing on one is sufficient),

$$\frac{\partial F(Q,K)}{\partial Q} = \frac{f'(Q)f(K)QK - f(Q)f(K)K}{(QK)^2} > 0$$

Since K is greater than 0, if $f(K) > 0$, this requires $f'(Q)Q - f(Q) > 0$, meaning $(\ln f(Q) - \ln Q)' > 0$. Classifying Q, it is required that $\ln f(Q) - \ln Q > c$, i.e., $f(Q) > cQ$. To ensure the derivative of the ratio is not a constant, $f(Q)$ should not be a linear function of Q. Therefore, it is required that for $Q \geq 1$, there exists $n > 0$ such that $f(Q) \geq cQ^{1+\frac{1}{n}}$. For $Q < 1$, there exists $n > 0$ such that $f(Q) > cQ^{1-\frac{1}{n}}$. For K less than 0, applying a negative sign to both sides of the inequalities leads to the same conclusions, with the signs reversed. The parameter n controls the degree of stretching, where smaller n results in more significant stretching, effectively converging to exponential functions like e^x as $n \to 0$.

It can be verified that when both Q and K are greater than 1,

$$\left(\frac{Q^{1+\frac{1}{n}}K^{1+\frac{1}{n}}}{QK}\right) = (QK)^{\frac{1}{n}}$$

and when both are less than 1,

$$\left(\frac{Q^{1-\frac{1}{n}}K^{1-\frac{1}{n}}}{QK}\right) = \left(\frac{1}{QK}\right)^{\frac{1}{n}}$$

For $Q > 1$ and $K < 1$,

$$\left(\frac{Q^{1+\frac{1}{n}}K^{1-\frac{1}{n}}}{QK}\right) = \left(\frac{Q}{K}\right)^{\frac{1}{n}}$$

demonstrating stretching in all cases.

However, the sufficient condition requires stretching with respect to QK, so it is necessary to verify that for $Q_1K_1 > Q_2K_2$ and $Q_1 > Q_2, K_1 < K_2$, the condition still holds. In other words, it is required that

$$\frac{f(Q_1)f(K_1)}{f(Q_2)f(K_2)} > \frac{Q_1K_1}{Q_2K_2}$$

which can be expressed as

$$\left(\frac{Q_1 K_1}{Q_2 K_2}\right)^{\frac{1}{n}} > 1$$

(when $Q_1 > Q_2 \geq 1, 1 \leq K_1 < K_2$). This condition is consistent with the previous requirements, ensuring that the sufficient condition holds. Similarly, this conclusion holds for different ranges of Q_1, K_1, Q_2, K_2, demonstrating internal consistency.

A.6. Piecewise Polynomial Theory

Below is a brief explanation of the piecewise polynomial theory.

Firstly, based on the definition of real tasks and category theory or set theory, it can be understood that all tasks can be completed by modeling as a function. The essence of neural network learning is to continuously fit this function using existing input and output through gradient iteration or other machine learning algorithms. So, what is an effective way to fit a function? Polynomial, bump function, or other base functions. Here, I adopt the theory of piecewise polynomials to analyze the function represented by neural networks. In fact, any function can be approximated by piecewise polynomials. For example, for any function, it can be divided into many segments of continuous functions based on its breakpoints, and each segment of continuous function can be approximated by taking a finite number of terms of the polynomial after Taylor expansion.

Next, a simple analysis of the function represented by deep neural networks. Because deep neural networks are computed recursively in a layer-by-layer manner, they can be obtained by using the recursive method of $Y = f(X) = f(x_1, x_2, ...) = f_n(...f_2(f_1(x_1, x_2, ...)))$. First, consider a single layer represented by f_i, where the MLP, CONV, Attention, and other operators are used to establish the manifold required for the task of fitting the data distribution. The activation function is used to implement piecewise-like functions (activation is akin to creating a breakpoint in the domain by discarding certain values).

Furthermore, with each layer of recursion, there are more breakpoints, and the frequency of breakpoints corresponds to the information frequency, i.e., deep layers represent high-frequency information, while shallow layers represent low-frequency information. If there is an ascending operation in each layer, then the deep layer will contain high-frequency information of high power of x^p (such as deep layers in images represent specific objects, while shallow layers represent low-frequency features such as edges and corners). Combined with the theory of piecewise polynomials, the multi-order fitting capability is definitely not lower than the first-order linear fitting.

According to the above analysis, it can explain or guide many things. Additionally, there may be errors in the analysis, for which we apologize in advance.

The necessity of piecewise, which can enhance the fitting ability, expand the function space that can be fitted, and alleviate overfitting.

The necessity of shortcuts, deep layers need to add low-order polynomials or low-frequency information from shallow layers to build a larger function space.

The selection of (depth, width, dim), it can be seen that they need to be relatively balanced to build a larger function space, fit more functions, but specific tasks may have biases and require an unbalanced setting.

Transformers use higher-order Attention operations, which approximate third-order polynomials and have stronger fitting capabilities than the first-order operations of CONV. However, there are differences in the capabilities of different third-order operations, so it is necessary to design necessary high-order interactions between different x_{ij} based on the task inputs. Moreover, higher-order operations are prone to overfitting and require more data.

Furthermore, it can explain emergence, where models of different depths represent the same object, such as x^{p_1} and x^{p_2} being equal to c; then, the granularity of different layers is c^{1/p_1} and c^{1/p_2}. Combined with knowledge from physics, a large number of simple combinations at a more microscopic level can explain very complex macroscopic problems. Therefore, the simple addition and multiplication of elements in deeper networks may emerge capabilities that are incomprehensible at the macroscopic level.

Of course, the above analysis is too simple to explain complex networks in reality, and there may also be errors. Due to

space constraints, we will conduct more complex and deeper modeling and experiments in future work.

A.7. Complexity Analysis

First, the partitioning of the Q region is conducted as a preprocessing step: Q tokens are classified using a classification matrix, or in cases where the sequence exhibits strong regional characteristics, no classification is performed. After preprocessing, the Q region undergoes segmentation, with the number of blocks approximately equal to the dimension size. Consequently, the size of each block is around $\frac{N}{dim}$. This segmentation is illustrated by the blue line box in Figure 3. For each region, corresponding KV tokens are selected, with each region choosing $\frac{CN}{dim}$ tokens. Generally, C is chosen as a small integer; in the experiments presented in this paper, C is set to 1. Thus, the selected KV tokens' length becomes $\frac{CN}{dim}dim$. Although the initially partitioned regions are relatively large, making the shared region not too small is important. However, experimentally, a constant multiple of the region size proves to be entirely sufficient. The complexity of the selection algorithm after the initial partitioning involves the computation of the dot product between a tensor with a shape of $(dim, mean(\frac{N}{dim}) = 1, dim)$ and a tensor with a shape of $(1, N, dim)$. The computational complexity of this operation is $Ndim^2$, and the shape of the selected KV tokens transforms into $(dim, \frac{CN}{dim}, dim)$.Next, a second partitioning is applied to Q, as depicted by the black line box in the above figure. This involves further partitioning the N/dim-sized regions of Q into s blocks, where each block has a size of N/dim/s. Subsequently, the second round of selection is performed based on this partitioning, resulting in the reshaping of the Q region representation to $(dim, s, mean(\frac{N}{s*dim}) = 1, dim)$. The shape of the KV from the previous selection is $(dim, 1, \frac{CN}{dim}, dim)$. The computational complexity of this operation involves the dot product of these two shapes, yielding $dim * s * \frac{CN}{dim}dim = CNdim^2$.

Recursively applying the aforementioned partitioning allows for reducing the size of the Q region to 1 after $log_d im(N)$ iterations. However, in practice, the size of the Q region is much larger than 1. Effectively utilizing multiple similar Q tokens to share the selected KV tokens is an efficient and effective approach. One can imagine that each Q token has closely related KV tokens, and similar Q tokens should select similar KV tokens. Therefore, sharing the selected KV tokens among these similar Q tokens is an effective and efficient strategy. Consequently, the final size of the partitioned Q regions is significantly larger than 1, and the actual partitioning constant is typically very small, usually below 3. Additional iterations beyond this point do not yield improved results. For instance, for a sequence length of 12K with dim=32, 2 iterations are sufficient. For moderate token lengths, such as 3K, typically one round of region block partitioning is adequate. Moreover, the number of partitions does not necessarily have to be precisely equal to dim; a small constant multiple of dim can be used. For KV tokens, if they exhibit strong regional characteristics, their region partitioning can be employed as selected tokens, similar to the Biformer approach, further reducing complexity.Each selection process described above can leverage the multiscale compression selection introduced in Section 2.1. In our experiments, we only applied multiscale compression selection in the final step of token selection.

Furthermore, after the extraction process, if the size of the extracted region is larger than dim, the following strategy is employed: since tokens are highly correlated, based on the analysis in Section 2.2, using a positive activation function combined with swapping the order to the power of p incurs minimal performance loss. Therefore, the final output is calculated using f(Q)(f(K)V). If the number of tokens extracted for the final region corresponding to Q is less than dim or around dim, Softmax, other correlation-based kernel functions, or piecewise polynomial functions can be used.

Because the shape of the finally selected KV tokens is $(s_0, s_1, ..., \frac{CN}{(s_0 s_1 ...)}, dim)$, using the order-swapping calculation method incurs a complexity of $s_0 s_1 ... \frac{CN}{(s_0 s_1 ...)}dim^2 = CNdim^2$. Subsequently, $s_0 s_1 ...dim^2$ matrices are output, with each dim^2 matrix responsible for a region in Q. The complexity for each matrix is shaped as $(s_0, s_1, ..., \frac{CN}{(s_0 s_1 ...)}, dim)$ for Q multiplied by the generated tensor with a shape of $(s_0, s_1, ..., dim, dim)$, resulting in a matrix multiplication complexity of $s_0 s_1 ... \frac{CN}{(s_0 s_1 ...)}dim^2 = Ndim^2$. It is evident that this approach is more efficient than the conventional method of order swapping, generating a single dim^2 responsible for the entire Q sequence. Similarly, when the number of tokens extracted for each region is less than dim, the complexity is$O(\frac{(CN)^2}{(s_0 s_1 ...)}dim)$, where $\frac{CN}{(s_0 s_1 ...)} \leq dim$. Therefore, the final computational complexity of this improved Attention mechanism is $min(O(CNdim^2), O(\frac{(CN)^2}{(s_0 s_1 ...)}dim))$, achieving linear complexity with respect to the sequence length N. Finally, if the model adopts dimensionality reduction and the sequence length in subsequent layers falls below dim, the original Attention strategy is employed. If efficiency reduction is acceptable, the selection mechanism can also be used.