Modified Attention with Non-Linear Kernels and its Impact on Few-Shot Learning

Jake Williams
williamsjl@uchicago.edu
University of Chicago
Chicago, IL, USA

Bayard Walsh
bkwalsh@uchicago.edu
University of Chicago
Chicago, IL, USA

Chenfeng Li
cfli@uchicago.edu
University of Chicago
Chicago, IL, USA

ABSTRACT

The attention mechanism in transformers is centered around the interaction between three learned components. We replace the traditional dot product interaction with functions developed for kernel methods and fine tune models designed for natural language processing. We study these new transformers on different few-shot learning tasks designed to test different abilities of the models.

KEYWORDS

Transformers, Deep Learning, Kernel Methods, Few-Shot Learning

ACM Reference Format:
Jake Williams, Bayard Walsh, and Chenfeng Li. . Modified Attention with Non-Linear Kernels and its Impact on Few-Shot Learning. In *Proceedings of University of Chicago (LLM Class).* ACM, New York, NY, USA, 9 pages.

1 INTRODUCTION

The modern state of the art in natural language processing is the transformer model. Transformers are deep learning systems that convert sequences of tokens into sequences of tokens, with both sets of tokens usually representing text, and are thus frequently referred to as large language models. In this paper, we will explore transformer models by modifying the traditional architecture using kernel methods and study the impact on few-shot learning.

We begin with a background on large language models and few-shot learning. Then, in section 3, we will explore the attention mechanism in detail. We will then briefly review the concepts of kernel methods, before considering how to modify attention using these concepts. We will give details as to how we have implemented the modified attention mechanism.

Then, in section 4, we will explore the concepts of few-shot learning, a common way of using large language models. We have selected a set of benchmark few-shot learning tasks which we will use to evaluate the modifications to the attention mechanism. The results of these tests will be presented in section 5.

2 BACKGROUND

Natural Language Processing (NLP) involves training machines to understand and interpret human language. In NLP, sentences are typically treated as sequences of tokens, with the objective for machines to predict subsequent tokens based on existing ones. Early language models primarily utilized Recurrent Neural Networks (RNNs) within an encoder-decoder, seq2seq architecture, transforming one sequence into another [22]. RNNs feature a series of hidden layers, each accepting a new token and a hidden vector from the previous cell to generate a new hidden vector. Enhancements to this structure, such as Long Short-Term Memory (LSTM) networks [10] and Gated Recurrent Units (GRUs) [5], introduced gating mechanisms to better maintain information over longer sequences.

A pivotal advancement came in 2017 when A. Vaswani et al. from Google Brain introduced the Transformer model, emphasizing the importance of the self-attention mechanism in understanding sentence semantics [23]. Unlike RNN-based models, the Transformer's encoder and decoder consist of several multi-head attention layers. Here, the attention of a query pays to a database of key-value pairs is computed as the weighted sum of the values, with weights derived using scaled dot-product attention. This model architecture facilitates increased parallelization during training, significantly reducing training times and marking a departure from the sequential processing of RNNs. Consequently, the development of large language models accelerated.

In 2018, A. Radford et al. from OpenAI introduced the Generative Pre-trained Transformer (GPT) model, unveiling the first version, GPT-1 [19]. In contrast to the original Transformer architecture, GPT models utilize only the decoder component, omitting the encoder. This series of models proved capable of addressing various language understanding tasks. OpenAI continued to scale up the models' size, with the number of parameters in GPT-1, GPT-2, and GPT-3 growing from 0.1 billion, 1.5 billion, to 175 billion, respectively [2, 20]. It's estimated that GPT-4 has around 1.76 trillion parameters [21]. Correspondingly, there have been significant improvements in model performance.

A notable development in 2020 was introduced by T. B. Brown et al., detailing the few-shot learning capabilities of GPT-3 [2]. As obtaining a large, labeled training dataset is impractical most of the time, GPT-3's few-shot learning enables it to perform NLP tasks using minimal data (such as brief prompts), leveraging its extensive pre-training. This advancement, coupled with the increased model size, allowed GPT-3 to handle tasks with high efficiency, often without the need for the extensive task-specific fine-tuning required by earlier models.

A significant paper on Machine Translation was published by Wenxiang Jiao et al., in 2023 [12], which aimed to analyze the capacity that GPT-4 had for language translation. The researchers discovered that GPT-3 preformed similarly to Google translate for "high-resource" languages, however struggled with translations to more distant languages. The paper mentions that GPT-4 shows improvement in translation across the board, and applied "pivot-prompting" or translating from the source sentence into a high-resource pivot language before into the target language, which greatly improved performance with GPT-4.

LLM Class, December 6, 2023, Chicago, IL
.

3 MODIFYING ATTENTION WITH KERNELS

At the core of transformers, the most successful architecture for natural language processing tasks today, is the attention mechanism. Our goal is to modify the traditional attention mechanism using concepts from kernel methods.

3.1 Kernel Methods

To begin discussing why kernel methods are useful in machine learning, we first define a kernel function.

Definition 3.1 (Kernel Function)**.** A kernel function is a mapping $k(\mathbf{x}, \mathbf{x}') \to \mathbb{R}$ which can be expressed as the inner product $\langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle$ for some Φ mapping into a dot product space.

Kernel functions, also known as covariance functions, are often used as a measure of similarity between two inputs [8]. These inputs can in theory be anything, including graphs or images, however we will only be considering vectors in this paper. Kernel functions are particularly useful in machine learning because they are positive semi-definite functions, meaning that any Gram matrix $K_{ij} = k(\mathbf{x_i}, \mathbf{x_j})$ is a positive semi-definite matrix [11]. Many learning algorithms rely on positive semi-definite matrices, such as least-squares or support vector machines (SVMs), though kernel methods are most closely connected to SVMs in practical usage.

In a support vector machine, the Gram matrix is constructed for all of the training data points and then used to learn weights corresponding to each training point. Then, to classify a new point, the kernel is taken between a new point and each training point, and the classification given by a weighted sum of these kernel values using the previously learned weights. Intuitively, the kernel function acts as a similarity between two inputs. In the first step, the similarity between each training point is used to learn how to classify points which are similar to any given training points. Then, at classification, we find which training points a new point is similar to and classify accordingly.

While considering the kernel function as a measure of similarity is appealing intuitively, it can also be viewed through the lens of definition 3.1. The kernel function acts as a dot product between two high dimensional feature vectors of the inputs, as defined by Φ. This allows us to lean on the well developed theory and algorithms which exist for linear methods while having the benefit of high dimensional non-linear featurization [11]. Even when the dimensionality of the feature vectors is infinite, we can often efficiently calculate the corresponding kernel, such as in the case of the Gaussian kernel.

The simplest kernel for two vector inputs is the linear kernel, which is simply the dot product between the two inputs. However, more complicated kernels can capture a wide variety of interactions, including periodic and other non-linear terms [8]. Kernel methods have been used directly in deep learning systems, and specific kernels have been designed to mimic the effects of deep learning [4]. The choice of kernel function for a given problem is difficult and important, so requires careful consideration [8].

3.2 Attention

The attention mechanism was designed for sequence based deep learning. In this setting, a sequence of tokens $t_1, \ldots, t_n$ is given, each with a set of corresponding vectors called keys, $\mathbf{k_i}$, queries, $\mathbf{q_i}$, and values, $\mathbf{v_i}$ [23]. For token i, its output from the attention mechanism is given as the following calculation:

(1) Construct a vector $\mathbf{s}$, where $s_j = \mathbf{q}_i \cdot \mathbf{k}_j$.
(2) Set $\mathbf{s}$ equal to its own softmax to normalize it.
(3) Set the output equal to $\mathbf{s} \cdot V$, where V is the collection of all value vectors into a matrix.

Originally, attention was designed as part of recurrent neural networks [1], however, recent advances have come from the introduction of the Transformer, which uses only the attention mechanism and standard neural network practices [23]. Transformers are now the state of the art in a wide variety of natural language processing tasks [17]

At the core of the attention mechanism is the dot product between queries and keys. In *Attention Is All You Need* [23], the result is described as the "compatibility" function, with an alternative from previous methods discussed. In this interpretation, the compatibility function determines how compatible keys are with a given query, and uses that to weight the corresponding values. This interpretation appeals to our intuition regarding kernel methods as measuring the similarity between two inputs, which is then used to weight another dot product. Therefore, we propose to consider alternative compatibility functions by considering commonly used kernel functions.

Note that the current attention mechanism already uses the linear kernel. In its place, we will test three alternatives:

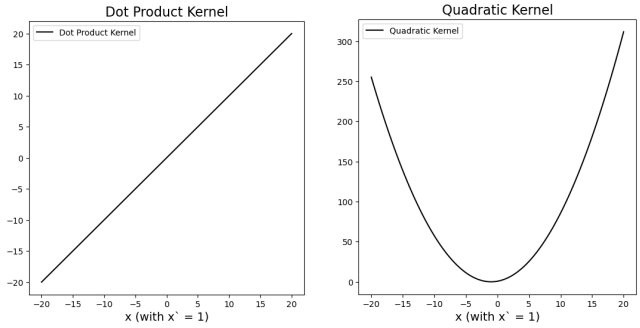
(1) The quadratic kernel, $k(\mathbf{x_1}, \mathbf{x_2}) = (\mathbf{x_1} \cdot \mathbf{x_2} + 1)^2$.
(2) The Gaussian kernel, $k(\mathbf{x_1}, \mathbf{x_2}) = e^{-\frac{|\mathbf{x_1} - \mathbf{x_2}|}{8}}$.
(3) The periodic kernel, $k(\mathbf{x_1}, \mathbf{x_2}) = e^{-2\sin(\frac{|\mathbf{x_1} - \mathbf{x_2}|}{10})}$

These were chosen from some of the more common kernel functions, and motivated in part by potential improvements in performance we foresee, as discussed in Section 4. Specifically, as the visualizations in Figure 1 shows, these kernels all introduce non-linearities, with the Gaussian and periodic kernels introducing unique relationships between points in a given space. The space here is that of embedded tokens, and we hope that these unique relationships are able to take advantage of properties of embedded tokens, such as similar meaning words being close in the embedded space, or additive vectors representing operations between words in the embedded space. We also give a brief discussion of the choices of hyperparameters for these kernels in the appendix section A.1.

For each choice of new kernel, we will fine-tune a GPT-2 model using OpenWebText [16]. We modified code from an existing code base called nanoGPT, which was meant for training and fine-tuning Transformers [13]. Full details of the training runs are given in appendix section A.2

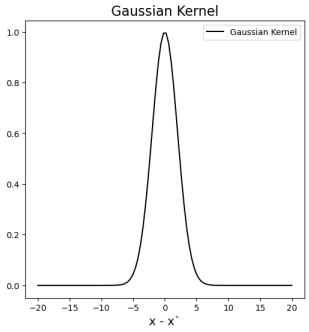
4 FEW-SHOT LEARNING

Our testing methodology applies the Few-Shot learning paradigm to measure the performance of our modified kernel in comparison to the baseline of the standard dot product. Few-Shot Learning is a supervised testing approach where a generalized pre-trained model is tested on a specific task after a limited amount of fine-tuning the model for that task [24]. This evaluation approach has been the key evaluation strategy applied to measure improvements in

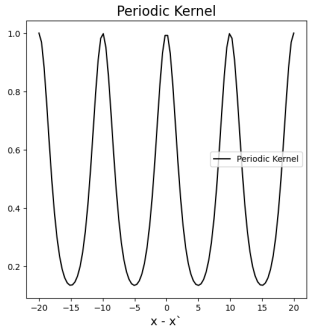


(a) Dot Product

(b) Quadratic Kernel



(c) Gaussian Kernel



(d) Periodic Kernel

Figure 1: Visualizations of the kernels used. In Figure 1a, the traditional linear kernel is shown. In Figure 1b, we see the effects of a simple non-linearity. Figure 1c shows how the Gaussian kernel changes the similarity to relying on the scale of either input to only the distance between the inputs, with the similarity being highest when they are close and decaying quickly further away. Lastly, Figure 1d shows the periodic kernel, which similarly relies on the distance between inputs, but periodically changes between high and low similarity.

GPT models; for example, in GPT-4 few-shot prompting was used for evaluating all testing benchmarks [17]. Because our model is a fine-tuned version of GPT-2 and we evaluate the model through a variety of benchmarks beyond text generation error rate, we decided to use this approach for our testing.

For evaluating performance benchmarks for our model, we selected three primary benchmarks, Massive Multitask Language Understanding (MMLU), Language Translation (English to French), and AI2 Reasoning Challenge (ARC). We believe that fine-tuning with different kernels may cause improved performance at different forms of evaluation due to the shapes of the kernel, therefore we decided to choose a variety of benchmarks for each of our kernels.

4.1 MMLU

This evaluation task considers a variety of knowledge, ranging from 57 academic subjects in multiple choice question format. We predict that the subjects that specifically rely on memorization such as High School Environmental Science or Biology may benefit from a narrow Gaussian kernel which would emphasize the most similar key-query pairs and therefore increase the model's capacity to recall memorized information by emphasizing the value of certain terms in the attention calculation. Because of the variety of tasks in the MMLU this evaluation model will measure to see if any of the kernels are well suited for any specific academic subject.

In terms of calculating a score on MMLU, we formatted the prompts to the model so that it selected one of the four options for the multiple-choice question. From there we measured the performance in terms of correct and incorrect answers on the tests. [9]

4.2 ARC

Similar to the MMLU evaluation metric, the AI2 Reasoning Challenge is a series of multiple-choice questions designed as a benchmark evaluator for research in advanced question-answering published by the Allen Institute for AI.

However, unlike MMLU, ARC explicitly tests logic and reasoning, meaning that memorization will be less critical thinking ability of the function will be tested. We theorize that the periodic kernel function may help the model understand patterns in logical questions, which will improve performance. [6]

4.3 Translation

Another evaluation bench mark utilized in for assessing model performance is translation from English to French. For this specific application, we used the BLEU translation equation as a method to measure the accuracy of the translation, which ranged from 0 to 100 based on the quality of the translation generated to the original text. We believe that translating text from English to French may benefit from a periodic kernel, which would emphasize the connection between translation in the embedding space and similar meaning of tokens. We used a sample of 40964 instances from the English-French Translation Data set on Kaggle. [7]

4.4 Training Loss

In addition to the evaluation scripts testing for specific model performance, we will also consider general text generation training loss while fine-tuning. However, because we are fine tuning the kernels on a pre-loaded linear model, training loss will most likely be far lower for the linear kernel.

5 RESULTS

In this section, we detail the results of testing the models described above on the set of benchmarks selected.

5.1 MMLU

The results for the MMLU test are given in Table 5.1. All four models do quite poorly given the circumstances. Note that in contrast to the results we will see for the ARC dataset in section 5.2, this test accepts only answers of A, B, C, or D. This means that there is no ability to choose a correct answer demonstrated by any of the four models. We did not fine tune the models to this specific task, which would most likely be necessary to remove the restriction on accepted outputs.

	Overall Accuracy	Maximum Category Accuracy	Minimum Category Accuracy
Baseline	0.24	0.34	0.15
Polynomial	0.24	0.33	0.15
Periodic	0.25	0.37	0.15
Gaussian	0.25	0.35	0.17

Table 1: Performance of the four model configurations on the MMLU test, separated by overall accuracy, highest accuracy on any one category, and lowest accuracy on any one category.

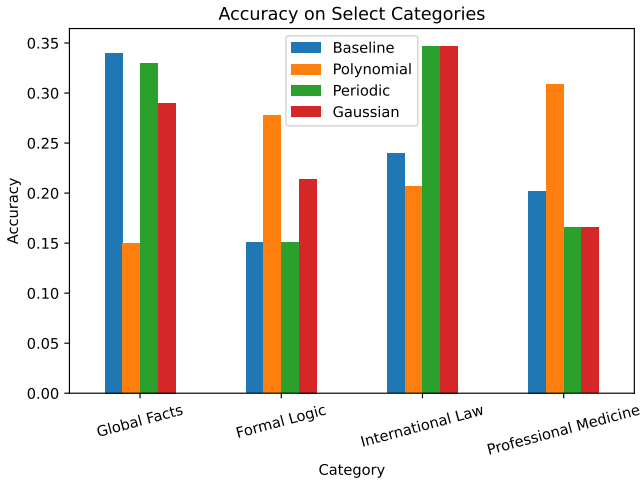


Figure 2: Accuracy of each model on select categories of the MMLU benchmark.

Even though the results demonstrate little to no ability to choose answers, there is still interesting information in the results obtained here. Specifically, in Table 5.1, we see the correlation between accuracy in each category across the four models. What stands out is that the Gaussian and periodic kernels are very highly correlated with regards to their accuracy in each category, and that the polynomial kernel is negatively correlated with all other models. This suggests that there are major differences between the information processing of the polynomial model and any other model, but that the Gaussian and periodic models are extremely similar. This makes sense if we consider that the periodic kernel is nearly identical to the Gaussian kernel in the range around 0 (see Figure 1).

Our hope in selecting different kernels is that each kernel will perform better for select types of reasoning or analysis. In Figure 2, we show each models' accuracy on 4 of the categories from the MMLU: Global Facts (the category on which the baseline accuracy was highest), Formal Logic (the category on which the baseline accuracy was lowest), International Law (the category on which the Gaussian accuracy was highest) and Professional Medicine (the category on which the Gaussian accuracy was lowest). Global Facts was also the category on which the polynomial accuracy was the lowest. We again see the similarity between the periodic and Gaussian models, as well as the stark difference of the polynomial accuracy. It's difficult to put too much stock into the categories which are successful or unsuccessful here given that the numbers are so low, but we do see that the categories with more memorization of facts - Global Facts and International Law - are the ones in which the baseline and Gaussian are successful, and the reasoning based categories - Formal Logic and Professional Medicine - are ones where they struggle. The full break down of accuracy per category is given in the Appendix Section B.1 Table 6.

	Baseline	Polynomial	Periodic	Gaussian
Baseline	1.00	-0.16	0.20	0.21
Polynomial	-0.16	1.00	-0.16	-0.17
Periodic	0.20	-0.16	1.00	0.71
Gaussian	0.21	-0.17	0.71	1.00

Table 2: Correlation between accuracy in each category of the four models. The Periodic and Gaussian models are highly correlated, while the polynomial kernel is somewhat negatively correlated with all other models.

5.2 ARC

The ARC dataset consists of a corpus of more than 14 million science-related sentences and includes 4967 easy and 2534 challenging multiple-choice questions, each with four answer options (A, B, C, or D). We fine-tuned each model using the corpus and created prompts that included answer examples and questions. The models were then tasked to generate a single new token for each question, interpreted as their answer. Only responses that exactly matched the letter of the true answer (irrespective of letter case) were considered correct. The results of this evaluation are detailed in Table 5.2.

	ARC-Easy		ARC-Challenge	
	Correct	Accuracy	Correct	Accuracy
Baseline	1096	0.221	591	0.233
Polynomial	39	0.008	67	0.026
Periodic	425	0.086	168	0.066
Gaussian	520	0.105	220	0.087

Table 3: Performance of the four model configurations on the ARC Easy and Challenge tests by the number of correct answer and corresponding accuracy.

Upon analyzing the results, it becomes evident that none of the four models could reliably select the correct answer, as indicated by their overall low accuracies in both the easy and challenging categories. Notably, the Baseline model, while still underperforming, demonstrated accuracies of 22.1% and 23.3% on the ARC-Easy and ARC-Challenge subsets, respectively. These figures are marginally better but still close to what would be expected from random guessing. In contrast, the models employing Polynomial, Periodic, and Gaussian kernels performed even worse than random guessing, with the Polynomial model being the least accurate. These results correspond to the training loss of each model listed in 5.4.

A closer inspection of the models' generated tokens for each question revealed that the Baseline model generally understood the multiple-choice format, often providing answers within the given choices. However, the models with the alternative kernels frequently generated irrelevant tokens, deviating from the expected A, B, C, or D answers. This inconsistency in generating appropriate responses underscores the challenges faced by the Polynomial, Periodic, and Gaussian models in understanding and responding within the structured format of the ARC questions.

Interestingly, the performance of the Periodic and Gaussian models was similar, reflecting the parallel mentioned in the previous section about the similarity of these two kernels. This observation points to a potential underlying similarity in how these kernels process and interpret information, which could be a focal point for further investigation. The findings overall highlight a significant gap in the capabilities of these advanced models compared to the more traditional Baseline model, suggesting a need for further refinement and understanding of self-attention mechanisms in the context of complex language comprehension tasks like those presented in the ARC dataset.

5.3 Translation

	Average BLEU score	Maximum	Minimum
Baseline	2.1696	88.5207	0
Polynomial	2.1858	88.6212	0
Periodic	2.1881	88.6212	0
Gaussian	2.1872	88.6212	0

Table 4: Performance of the four model configurations across the BLEU metric

The Bilingual Evaluation Understudy or BLEU metric was the evaluation strategy applied to measure the models translation capabilities from English to French. After training, the model a testing set of 8193 overall translations for a variety of short phrases in English to French, and then calculated and averaged the BLEU score across these terms.

Across the board, the models performed very poorly, with an average BLEU score that would be considered "Almost useless" for most translations.[14] However note that each of the models were able to achieve a maximum of BLEU score of above 60, which marks a high quality translation.

The average and maximum BLEU scores for all of the models are very similar, indicating that the changes in the kernel made a negligible impact on the performance of the model with regard to translation. However it is notable that the Baseline model had the lowest performance for both the average and maximum BLEU score, if only by a slight amount. After fine tuning on Open Web Text, the Baseline model had the lowest training loss among the four kernels with 4.3813, suggesting it was far better at generating text than the other configurations, however this performance does not translate to English to French translation.

An explanation for this inconsistency could be the fine tuning with OpenWebText - this data is scrapped from Reddit comments and is largely a corpus of English text. In the 2023 machine translation paper by Wenxiang Jiao et al [12], their experiments on translation with GPT-3 performed worse on Reddit comments and better on European language translation. Therefore the GPT-2 model fine-tuned on Reddit data demonstrates the converse of this case, where its exposure to Reddit decreases its translation ability.

5.3.1 Limitations. A key limitation in this analysis was computation in terms of training data and fine tuning capabilities. We fine tuned our models from the pre-trained version of GPT2 for each kernel on the Open Web Text, and our initial strategy was to perform further task specific fine tuning for each model. However, because of the limitation for GPU access, we could only apply one epoch for fine tuning specific to the translation evaluation. Therefore our translations were essentially evaluated on a fine tuned version specific to each type of kernel, but not to each task- which explains in part why all of the models had bad results.

5.4 Open Web Text Training Error

Kernel	Train Loss
Baseline	4.3813
Quadratic	11.0376
Periodic	9.4852
Gaussian	9.3756

Table 5: Training loss for the four model configurations after fine tuning the pre-trained GPT-2 model on the Open Web Text data set.

Here we have the training error value given after fine tuning the pre-trained GPT-2 model on the Open Web Text data set. As expected, the baseline kernel had the smallest training loss by a considerable margin. This is likely due to the small period of fine-tuning not being enough time for the model to relearn the given kernel configuration.

Additionally, after task specific fine tuning, the training loss increased in every non-linear kernel case. This drop in training loss after additional fine tuning indicates that the strategy of switching kernel from a pretrained model will decrease text generation performance, meaning that in future experiments with non-linear kernels, it would be a better strategy to train models from scratch.

6 DISCUSSION

6.1 Run time

To consider the scalability of the proposed kernels, one of the main trade-offs in modifying the standard dot kernel would be the potential for increased run time from extra kernel arithmetic. Note that linear projection is used on Periodic and Gaussian kernels after the kernel is applied to resize the dimensions of the tensor for the model. We have run time for the given kernels, given the attention components q, k where n, h is the hidden size or number of hidden units, B is the batch size, T is the sequence length or time steps, and C is the input size or number of input features.:

Baseline:
Matrix multiplication: $q@k^{\mathrm{T}}(-2,-1) = O(B \cdot \mathrm{nh} \cdot T^2 \cdot C)$

Element-wise multiplication and division: $O(B \cdot \mathrm{nh} \cdot T^2 \cdot C)$

Linear projection: No linear projection is applied.

Polynomial Kernel:
Matrix multiplication: $q@k^{\mathrm{T}}(-2,-1) = O(B \cdot \mathrm{nh} \cdot T^2 \cdot C)$

Addition and exponentiation: c^d takes $O(1)$ operations.

Element-wise multiplication and division: $O(B \cdot \mathrm{nh} \cdot T^2 \cdot C)$

Linear projection: No linear projection is applied.

Periodic Kernel:
Trigonometric operations: $\sin\left(\frac{\pi \cdot (q-k)}{p}\right)^2 = O(B \cdot \mathrm{nh} \cdot T^2 \cdot C)$

Exponential and scalar operations: $O(B \cdot \mathrm{nh} \cdot T^2 \cdot C)$

Linear projection: $O(B \cdot \mathrm{nh} \cdot T^2 \cdot C)$

Gaussian Kernel:
Exponential and scalar operations: $\exp\left(\frac{(q-k)^2}{-2 \cdot l^2}\right) = O(B \cdot \mathrm{nh} \cdot T^2 \cdot C)$

Linear projection: $O(B \cdot \mathrm{nh} \cdot T^2 \cdot C)$

Therefore in generalized run time analysis, the dominating term for all kernel computations is $O(B \cdot \mathrm{nh} \cdot T^2 \cdot C)$, meaning that the kernels all have approximately the same run time as n and h scale larger and larger. However with the intention of large-scale implementation for the model, in practice, the dot product will be computed many times on a limited-size tensor. Therefore, with repeated computation on a matrix with a set bound for n and h, the extra computations for Gaussian and Periodic kernels would increase the run time.

Another factor to consider in run-time analysis is the cost of generating the standard deviation from the data set in question, which is used as a hyperparameter for the Gaussian kernel. During the experiment, we found $STD = 0.115$ for the Open Web Text, which was then manually inputted into the Gaussian for $p = 1/STD$. In a larger or incomplete data set, generating this value could be far more expensive and would be considered an additional cost to computation.

Furthermore, with the modified kernels, we must consider the specificity of the current GPU hardware tailored to certain types of computation. For example, the periodic kernel requires computing π in every kernel computation, and if a GPU is specifically designed for certain matrix arithmetic functions the cost of computing this variable could be higher than expected for larger training runs.

6.2 Kernels as Similarity Functions

As discussed in Section 3.2, the attention mechanism relies on the idea of a compatibility function between keys and queries [23]. Kernels are a natural pool of such functions to choose from, and provide a wide variety of archetypes, including non-linearity. However, choosing the right one can be difficult even in well established kernel learning problems [8]. Our results indicate that changing kernels in the attention mechanism can drastically change the functionality of the model. However, we have also found that some kernels are very similar, notably the Gaussian and periodic kernels behaving similarly. To make the Gaussian and periodic kernels function differently, we need to set their hyperparameters such that they give very different similarities on the same inputs, while still choosing values which reasonably fit the data. This would likely require more exploration with the model than we have resources for in this paper.

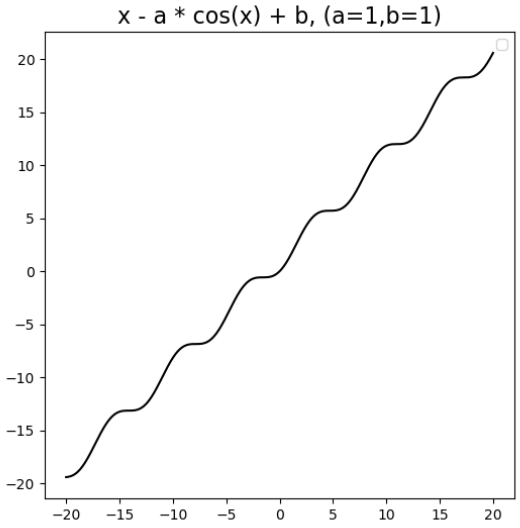
The differences between each kernel function highlight another potential shortcoming of our methodology, namely that we fine-tuned each model from GPT-2, which was trained entirely using the linear kernel. This could be problematic given the intuition that the kernel learns a similarity between keys and queries. If we start from a pre-trained model which produces keys and queries that are meant to be compared using the linear kernel, it may actually be more difficult to learn meaningful similarities with regards to a new kernel such as the Gaussian kernel. It may have been more beneficial to train the model from scratch to avoid needing to unlearn any kernel specific notions. Consider in particular the polynomial kernel, which is just the square of the linear kernel (plus a constant term). The linear kernel is designed to output large negative numbers for particular relationships, while the quadratic kernel we used will never output negative numbers at all, and will output large positive numbers for what would have been large negative relationships. This is a major change that likely requires more time to properly train than the fine-tuning start could get. It also helps to explain why the quadratic kernel behaves opposite the baseline in a large number of cases.

6.3 Future Work

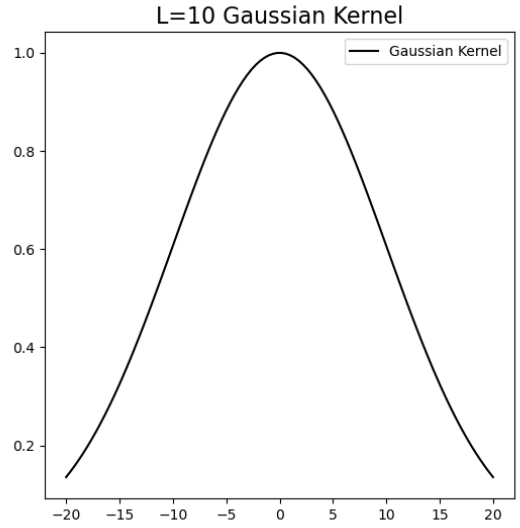
If we were to continue this project, we would likely modify many of the pracitices we selected. For starters, as mentioned above, we would prefer to train from scratch rather than starting from a point reached with a different kernel. We also would like to increase the size of the model, given that larger models have shown much greater success in Few-Shot Learning tasks [3].

Another option for future work is modifying the specific kernel function used from the three selected in this paper. As previously mentioned, our process of switching from the linear kernel to another kernel during fine-tuning makes evaluating the potential for alternative kernels difficult. However, in some of the examples generated by the periodic kernel, the model was repeating a short phrase several times; this could indicate that the attention kernel is drawing too much attention to a limited amount of word vectors. Therefore a mix of a linear and periodic kernel function could capture the potential benefits of periodicity while maintaining the attention structure.

For example, the $x - a * cos(x) + b$ kernel captures some linearity while adding a slight periodicity that may better adjust to language generation. The kernel's hype parameters also could be fine-tuned; perhaps a smaller model could be implemented to determine the



(a) Linear-Periodic Kernel



(b) Wide Gaussian Kernel

Figure 3: Visualizations of proposed kernels for future testing. In Figure 3a, the periodic kernel with linearity. In Figure 3b, we see the effects of a wider Gaussian kernel.

best fit for these values. However note that calculating a and b values specific to a certain shape of data or for a more narrow purpose could improve performance for the kernel tailored to a certain task, although it limits the general applicability of the model as well as could increase model run time depending on the calculations needed for a and b.

Along with the linear kernel, a wider Gaussian could also improve training performance. The Gaussian and periodic kernels exhibited similar behaviors, implying that the narrow focus near the origin greatly impacted the attention calculation more so than the periodic behavior outside range of $-5, 5$. Furthermore, the generated text included examples where phrases were repeated many times. Therefore, a wider quadratic function, as shown in figure 3b, could provide a wider distribution in the attention calculation and prevent the repetitious behavior the tested Gaussian kernel exhibited.

Lastly, we might want to consider new benchmarks and tasks which would fit the geometric intuition of the kernels more explicitly. With the language reasoning tasks we chose, we tried to think about how the embeddings of words might relate to each other geometrically in our choice of kernels. However, the Transformer architecture can be used for non-language based tasks, such as protein structure prediction [15] or particle transport [18], where geometric intuition may not only be more accurate and useful, but potentially necessary for physically accurate models.

7 CONCLUSION

Our results indicate that changing the attention mechanism using non-linear kernels is impactful, and changes the relationships between tokens meaningfully. However, our experiments were not sufficient to create models which used these changes well. Pitfalls we believe led to these shortcomings include attempting to fine-tune models trained with a different kernel, too small of models for few shot learning, and additional fitting of kernel shape and hyperparameters to the data. Given these potential changes, we see promise in new kernel functions as ways to knowledgeably impact the behavior of large language models.

ACKNOWLEDGMENTS

We extend our thanks to Xuefeng Liu, for training our models and helping us complete our project, as well as to Professors Stevens and Foster for the fascinating lectures.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2016. Neural Machine Translation by Jointly Learning to Align and Translate. arXiv:1409.0473 [cs.CL]

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, et al. 2020. Language Models are Few-Shot Learners. *Advances in Neural Information Processing Systems* 33 (2020), 1877–1901.

[3] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. arXiv:2005.14165 [cs.CL]

[4] Youngmin Cho and Lawrence Saul. 2009. Kernel Methods for Deep Learning. In *Advances in Neural Information Processing Systems*, Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta (Eds.), Vol. 22. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2009/file/5751ec3e9a4feab575962e78e006250d-Paper.pdf

[5] Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. arXiv:1412.3555 [cs.CL]

[6] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge. *CoRR* abs/1803.05457 (2018). arXiv:1803.05457 http://arxiv.org/abs/1803.05457

[7] Dhruvil Dave. 2021. English-French Translation Dataset. https://doi.org/10.34740/KAGGLE/DSV/1926230

[8] David Kristjanson Duvenaud. 2014. *Automatic Model Construction with Gaussian Processes.* PhD thesis. Pembroke College.

[9] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring Massive Multitask Language Understanding. *CoRR* abs/2009.03300 (2020). arXiv:2009.03300 https://arxiv.org/abs/2009.03300

[10] Sepp Hochreiter and Jürgen Schmidhuber. 1997. LONG SHORT-TERM MEMORY. *Neural computation* 9, 8 (1997), 1735–1780.

[11] Thomas Hofmann, Bernhard Schölkopf, and Alexander J. Smola. 2008. Kernel methods in machine learning. *The Annals of Statistics* 36, 3 (2008), 1171 – 1220. https://doi.org/10.1214/009053607000000677

[12] Wenxiang Jiao, Wenxuan Wang, Jen tse Huang, Xing Wang, Shuming Shi, and Zhaopeng Tu. 2023. Is ChatGPT A Good Translator? Yes With GPT-4 As The Engine. arXiv:2301.08745 [cs.CL]

[13] Andrej Karpathy. 2023. nanoGPT. https://github.com/karpathy/nanoGPT.

[14] Alon Lavie. 2010. Evaluating the Output of Machine Translation Systems. (01 2010).

[15] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Salvatore Candido, and Alexander Rives. 2023. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science* 379, 6637 (2023), 1123–1130. https://doi.org/10.1126/science.ade2574 arXiv:https://www.science.org/doi/pdf/10.1126/science.ade2574

[16] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv:1907.11692 [cs.CL]

[17] OpenAI. 2023. GPT-4 Technical Report. arXiv:2303.08774 [cs.CL]

[18] Oscar Pastor-Serrano and Zoltán Perkó. 2022. Learning the Physics of Particle Transport via Transformers. *Proceedings of the AAAI Conference on Artificial Intelligence* 36, 11 (Jun. 2022), 12071–12079. https://doi.org/10.1609/aaai.v36i11.21466

[19] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving Language Understanding by Generative Pre-Training. https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf.

[20] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners. *OpenAI Blog* 1, 8 (2019), 9.

[21] Maximilian Schreiner. 2023. GPT-4 Architecture, Datasets, Costs and More Leaked. https://the-decoder.com/gpt-4-architecture-datasets-costs-and-more-leaked/. Archived from the original on July 12, 2023. Retrieved July 12, 2023.

[22] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to Sequence Learning with Neural Networks. *Advances in Neural Information Processing Systems* 27 (2014).

[23] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention Is All You Need. arXiv:1706.03762 [cs.CL]

[24] Yaqing Wang and Quanming Yao. 2019. Few-shot Learning: A Survey. *CoRR* abs/1904.05046 (2019). arXiv:1904.05046 http://arxiv.org/abs/1904.05046

A HYPERPARAMETER SELECTION

Here we discuss various criteria for selecting hyperparameters in the models.

A.1 Kernels

Hyperparameters for each kernel were chosen before training, and no hyperparameter search was done. Generally, we aimed for simplicity where possible, setting most parameters to 1. We also chose to keep the polynomial kernel as a quadratic ($d = 2$) for the same reason. The main choices, therefore, were the denominators in the Gaussian and periodic kernels.

To select the denominators of the aforementioned kernels, we considered the distribution of values of the embedded tokens. Since the goal of the kernels is to capture new dependencies in this space, we would like the kernels to fit within this space reasonably. We calculated the mean and standard deviation of all values that the embedded tokens take on, and used these metrics to inform our decisions with regards to the hyperparameters.

To assess all values the embedded tokens take on, we iterated over every word in the OpenWebText [16] training dataset and used the GPT-2 tokenizer and first layer embedding function to obtain values for each token that appears. The mean value was found to be approximately zero and the standard deviation approximately 0.1. We chose the hyperparameters so that the Gaussian would be somewhat "tight" within this space and the periodic function would have multiple periods within a few standard deviations.

A.2 Training Runs

For the training runs, the model was initiated from a pre-trained version of GPT-2, then fine tuned on OpenWebText on a NVIDIA V100 Tensor Core GPU. This fine tuning process lasted 7h 8m. The following configurations were used to finetune the 4 kernels from the pre-trained model:

```
batch_size = 4
block_size = 1024
gradient_accumulation_steps = 5 * 4
max_iters = 12000
lr_decay_iters = 12000
eval_interval = 200
eval_iters = 200
log_interval = 10
weight_decay = 1e-1
```

Because of limitations on compute resources and time to train, evaluation parameters were greatly reduced. For fine tuning for specific evaluation tasks, the following configurations were used:

```
batch_size = 1
gradient_accumulation_steps = 1
max_iters = 1
learning_rate = 3e-5
eval_interval = 2
eval_iters = 2
```

B ADDITIONAL RESULTS

Here we present additional results obtained from the benchmarks.

B.1 Full MMLU Results

Here we present a table that includes the accuracy of all four models broken down into each category, with the highest accuracy in each category highlighted.

Table 6: Model accuracy broken down by category.

	Baseline	Polynomial	Periodic	Gaussian
Abstract Algebra	0.26	0.24	**0.28**	0.27
Anatomy	0.24	0.20	**0.33**	0.31
Astronomy	0.21	0.28	0.25	**0.29**
Business Ethics	0.26	**0.32**	0.22	0.23
Clinical Knowledge	0.23	0.21	**0.26**	0.23
College Biology	0.22	0.27	**0.28**	0.28
College Chemistry	0.20	**0.26**	0.22	0.20
College Comp Sci	0.26	**0.28**	0.27	0.25
College Math	0.28	**0.30**	0.24	0.20

College Medicine	0.22	**0.29**	0.24	0.26
College Physics	0.19	0.25	**0.26**	0.25
Computer Security	0.24	0.21	0.21	**0.26**
Conceptual Physics	**0.29**	0.23	0.24	0.23
Econometrics	0.20	0.22	0.18	**0.23**
Electrical Engineering	0.18	0.21	0.22	**0.27**
Elementary Math	**0.28**	0.24	0.28	0.26
Formal Logic	0.15	**0.28**	0.21	0.15
Global Facts	**0.34**	0.15	0.29	0.33
HS Biology	0.22	0.19	**0.24**	0.22
HS Chemistry	0.27	0.23	0.23	**0.27**
HS Comp Sci	0.28	0.20	**0.32**	0.29
HS European History	0.25	0.22	0.28	**0.29**
HS Geography	0.22	0.23	0.25	**0.25**
HS Gov & Politics	0.21	0.19	0.23	**0.23**
HS Macroeconomics	0.23	0.23	**0.24**	0.22
HS Mathematics	0.26	**0.28**	0.26	0.26
HS Microeconomics	0.20	**0.22**	0.21	0.21
HS Physics	0.25	0.26	0.23	**0.29**
HS Psychology	0.23	0.21	**0.23**	0.23
HS Statistics	**0.31**	0.17	0.22	0.21
HS Us History	0.23	0.22	**0.26**	0.26
HS World History	**0.30**	0.23	0.27	0.27
Human Aging	0.23	**0.33**	0.28	0.26
Human Sexuality	**0.27**	0.24	0.24	0.24
International Law	0.24	0.21	**0.35**	0.35
Jurisprudence	0.23	0.25	0.25	**0.30**
Logical Fallacies	0.28	0.26	0.27	**0.31**
Machine Learning	**0.26**	0.26	0.23	0.21
Management	0.24	0.23	**0.27**	0.19
Marketing	**0.29**	0.26	0.22	0.24
Medical Genetics	0.25	**0.28**	0.18	0.21
Miscellaneous	**0.29**	0.23	0.22	0.24
Moral Disputes	0.25	0.29	0.28	**0.30**
Moral Scenarios	0.24	0.24	0.24	**0.24**
Nutrition	0.21	0.24	0.25	**0.25**
Philosophy	0.24	0.18	**0.29**	0.29
Prehistory	0.26	0.24	**0.27**	0.26
Professional Accounting	0.26	0.23	0.26	**0.28**
Professional Law	0.24	0.26	**0.27**	0.26
Professional Medicine	0.20	**0.31**	0.17	0.17
Professional Psychology	0.26	0.24	0.27	**0.28**
Public Relations	**0.28**	0.25	0.25	0.25
Security Studies	0.18	0.20	**0.24**	0.24
Sociology	0.19	**0.27**	0.24	0.25
US Foreign Policy	0.21	0.27	**0.29**	0.25
Virology	**0.28**	0.27	0.21	0.20
World Religions	0.20	0.29	0.29	**0.37**

Received 6 December 2023