Building Vision Transformers with Performized Attention Kernals to Classify the MNIST Handwritten Numbers

Sara Davis
Soujanya Duggirala

Project Overview

Deep learning is a field of artificial intelligence that involves mimicking the structure of neurons in a human brain to automate tasks done by humans, such as reading and writing text, translating languages, and image classification. These models are unique in that nonlinearity can easily be introduced to their architectures through the use of activation functions. The next generation of deep learning is Transformers, which process information sequentially and with an activation function known as an attention kernel that weights data based on relevance to the information it is learning. Performer Transformers are a specialized architecture that estimates attention kernels, but, as opposed to traditional Transformers, has linear time and computational complexity, which is valuable for faster processing. Performers are also unique in that they are not dependent on prior low-rankness or sparseness, and utilize fast attention through positive orthogonal random features (Choromanski, 2022).

For Computer Vision applications, a common problem that is considered the "Hello World" problem of deep learning is image classification of handwritten numbers using the MNIST handwritten numbers dataset, collected by the National Institute of Standards and Technology. In this project, we build Vision Transformers and Performer Transformers to classify the MNIST handwritten numbers and test different Performer attention kernels to check for the training/testing speed and accuracy. We also run ablation studies over different randomly selected features in the dataset using redraw and no-redraw strategies.

Methodology

MNIST Dataset & Data Processing

The first step was to load the MNIST dataset into Python. The Modified National Institute of Standards and Technology dataset contains black and white handwritten digits, ranging from 0 through 9, of American Census Bureau employees and American high school students (Grother, 1996). The dataset contains 70,000 images total of the individual pixel colors in an array, and it is split with 60,000 images set aside for training and 10,000 images for testing. It is commonly used for image classification applications.

The images in both the testing and training sets of the MNIST data set are first flattened, meaning that they are each transformed into a single row vector. The image pixel values are then normalized to be in the range of 0-1 inclusive. The vector is then one-hot encoded to be suitable for categorization by the vision and performer transformers.

Performer Attention Functions

In implementation to keep the trials standardized, we used the same hyperparameters for all runs, most importantly with a learning rate of 0.01, a weight decay of 0.001, and 100 epochs of training. Overall, we assessed six different attention functions for this project. They are listed in Equations 1-6 below.

Equation 1: Softmax	$\sigma(z) = \frac{e^{z_i}}{\Sigma_{j=1}^{K} \quad e^{z_j}}$
Equation 2: ReLU	$f(x) = max(0, x)$
Equation 3: Performer-X4	$f(x) = x^4$
Equation 4: Performer-Quad	$f(x) = max(0, x^4)$
Equation 5: Quadratic	$f(x) = x^2$
Equation 6: Performer-X2	$f(x) = max(0, x^2)$

Evaluation Metric

We use the built-in evaluation method in Keras to assess the performance of each method from the Keras toolbox; this returns a scalar measure representing the test set loss. This was then converted to a percentage for reporting purposes.

Ablation Studies

We conducted ablation studies to investigate how training on smaller random samples (with and without replacement) from the total dataset (70,000 rows) changed the overall model performance and training time. Six trials total were conducted using 10,000, 20,000, or 50,000 samples for training and either with or without replacement.

Results & Evaluation

Table 1 shows the error rates and processing times for the Vision Transformer and the performized variants using the full dataset from MNIST. The error rates and processing times are also demonstrated in Figures 1 and 2 respectively. Overall, the performer variants perform better than the model with the softmax attention function, with the notable exception of Performer-ReLU, which did not perform well given the hyperparameters standardized across all attention functions. The error rate for the Performer-ReLU is very high; perhaps with additional training epochs, this configuration would have yielded better results.

For the ablation studies, the results are shown in Table 2 for sampling without replacement and in Table 3 for sampling with replacement for sample sizes of 10,000, 20,000, and 50,000 respectively. Their performance is also shown visually in Figures 3-6. We see that for both cases, the accuracy increases with the number of samples, with some exceptions that can possibly be attributed to random number generation. We can fix this in further work by specifying a seed for random number generation. The processing time also notably increases with more samples, which is expected. In terms of processing time, the x^2 attention function training was the quickest across almost all trials. Redrawing without replacement performed only slightly worse than redrawing with replacement with regards to the error rates.

Table 1: Baseline Performer Evaluation

Attention Function	**Error (%)**	**Processing Time (seconds)**
Softmax (not Performized)	6.09	48.33
Performer-ReLU	90.2	56.78
x^4	2.10	62.80
Performer-Quad ($\max(x^4,0)$)	3.47	63.38
x^2	3.16	46.92
$\max(x^2,0)$	3.85	48.58

Table 2: Redraw without Replacement Performer Evaluation

Performer Attention Function	**10,000 Samples**		**20,000 Samples**		**50,000 Samples**	
	Error (%)	**Processing Time (seconds)**	**Error (%)**	**Processing Time (seconds)**	**Error (%)**	**Processing Time (seconds)**
Softmax (not Performer)	8.47	9.65	7.28	16.90	6.59	41.88
ReLU	89.15	12.48	90.02	15.69	90.21	45.37
x^4	6.37	10.89	3.61	19.83	2.39	47.53
Quad	6.37	12.48	5.50	20.74	2.56	50.79
x^2	6.37	9.54	5.15	20.96	5.59	38.03
$\max(x^2,0)$	6.09	20.95	4.20	16.72	10.63	41.60

Table 3: Redraw with Replacement Performer Evaluation

Performer Attention Function	10,000 Samples		20,000 Samples		50,000 Samples	
	Error (%)	**Processing Time (seconds)**	**Error (%)**	**Processing Time (seconds)**	**Error (%)**	**Processing Time (seconds)**
Softmax (not Performer)	14.08	8.81	7.46	20.95	6.89	43.93
ReLU	88.52	9.16	90.23	16.26	90.07	38.50
x^4	3.71	10.35	5.64	20.97	3.14	48.24
Quad	7.07	10.86	3.36	20.45	3.96	82.42
x^2	7.42	9.58	9.10	20.95	4.16	40.70
max(x^2,0)	8.68	9.47	8.23	16.70	4.80	40.55

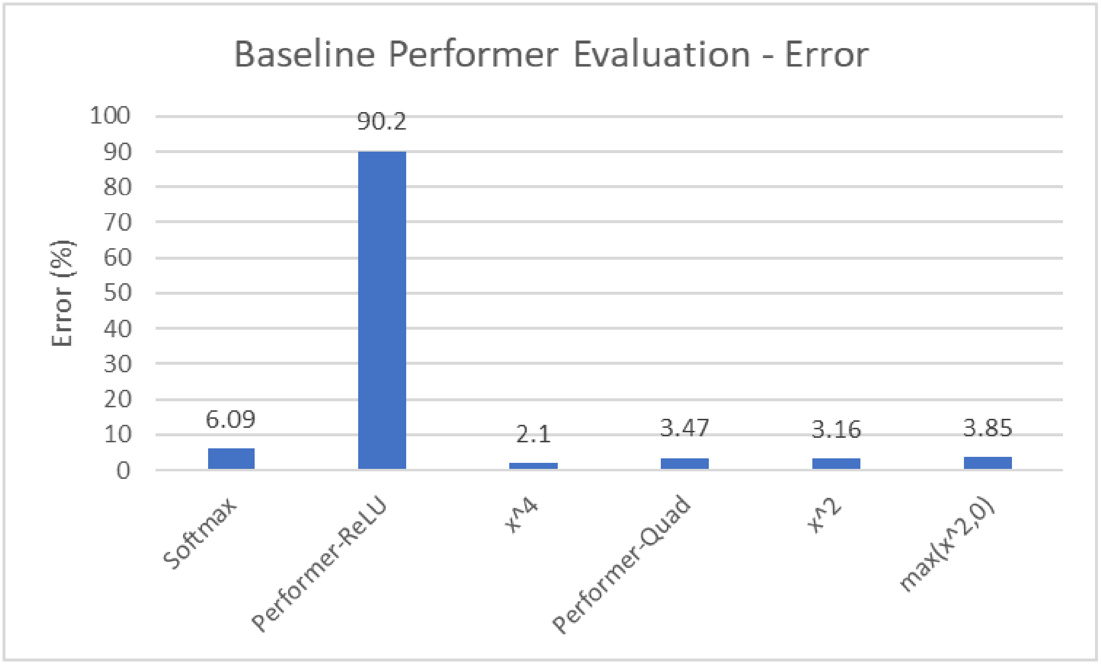

Figure 1: Baseline Performer Evaluation Error Rates.

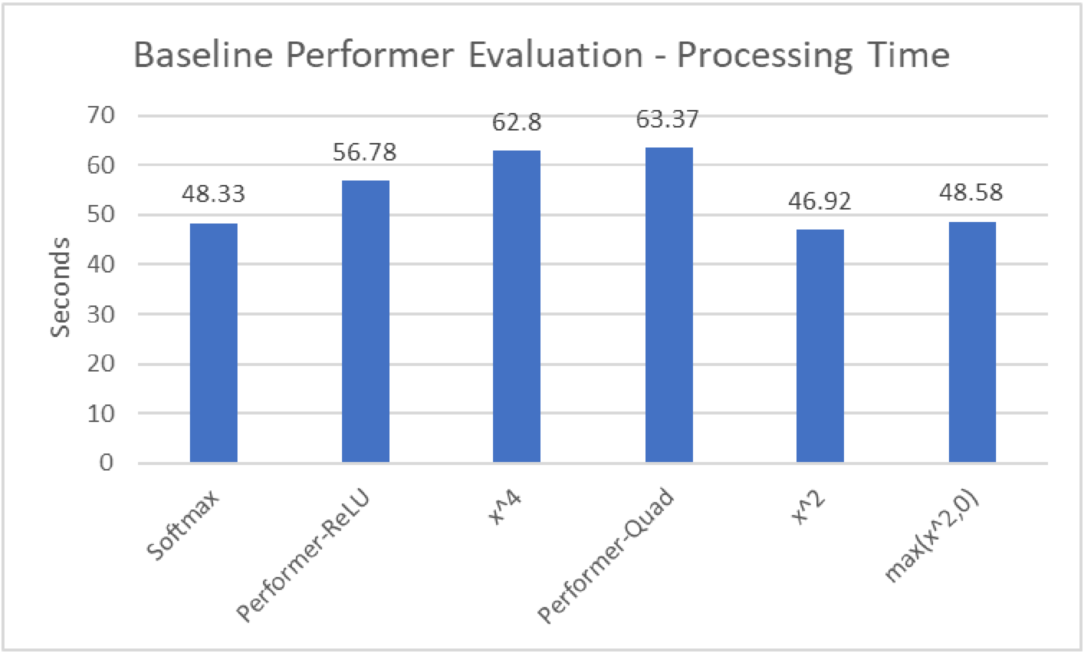

Figure 2: Baseline Performer Evaluation Processing Times.

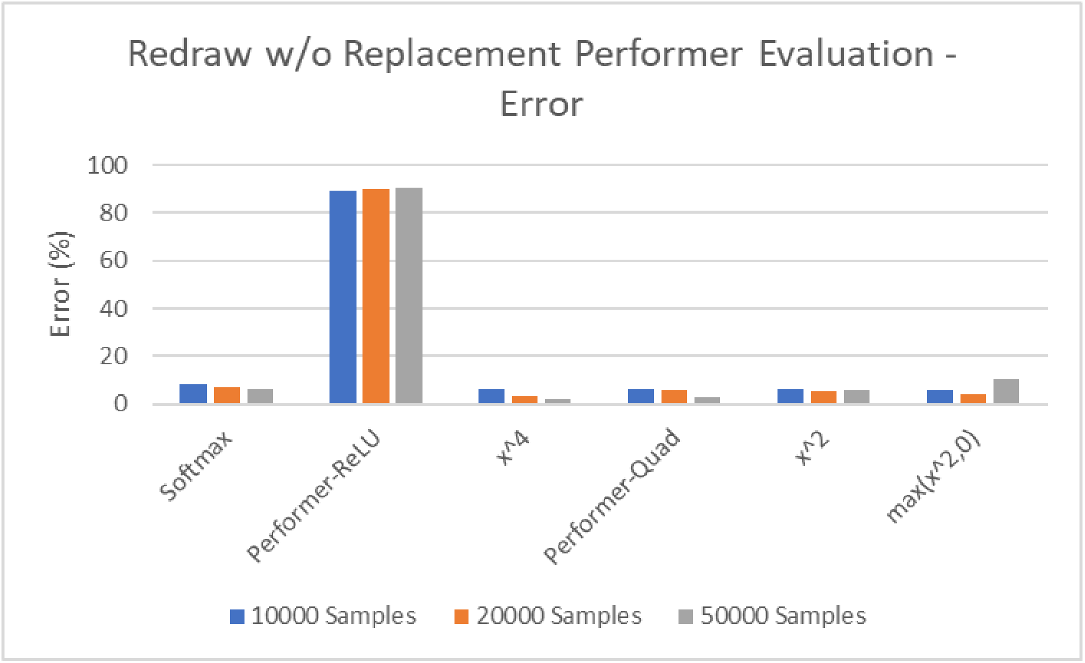

Figure 3: Redraw without Replacement Performer Evaluation Error Rates.

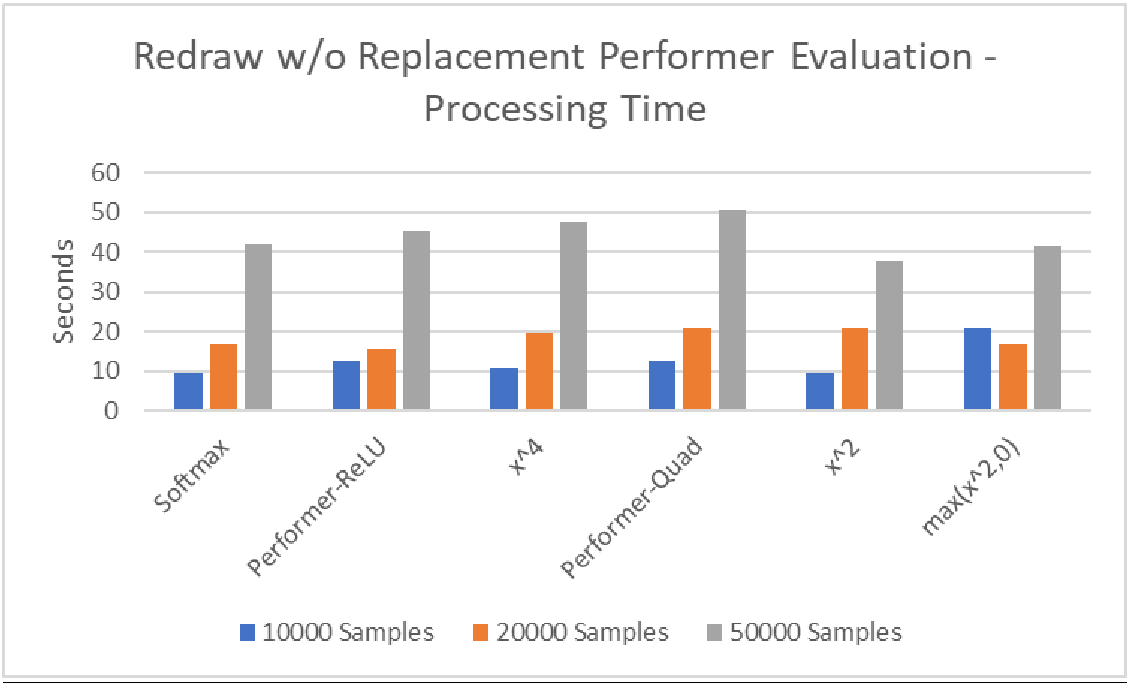

Figure 4: Redraw without Replacement Performer Processing Times.

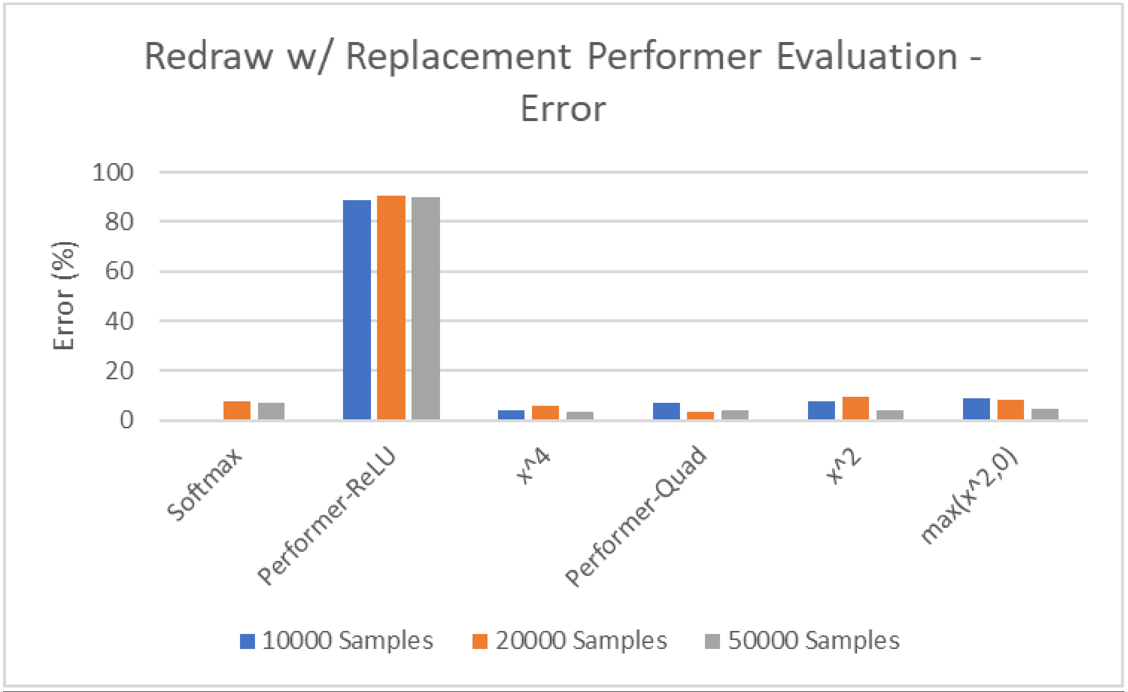

Figure 5: Redraw with Replacement Performer Evaluation Error Rates.

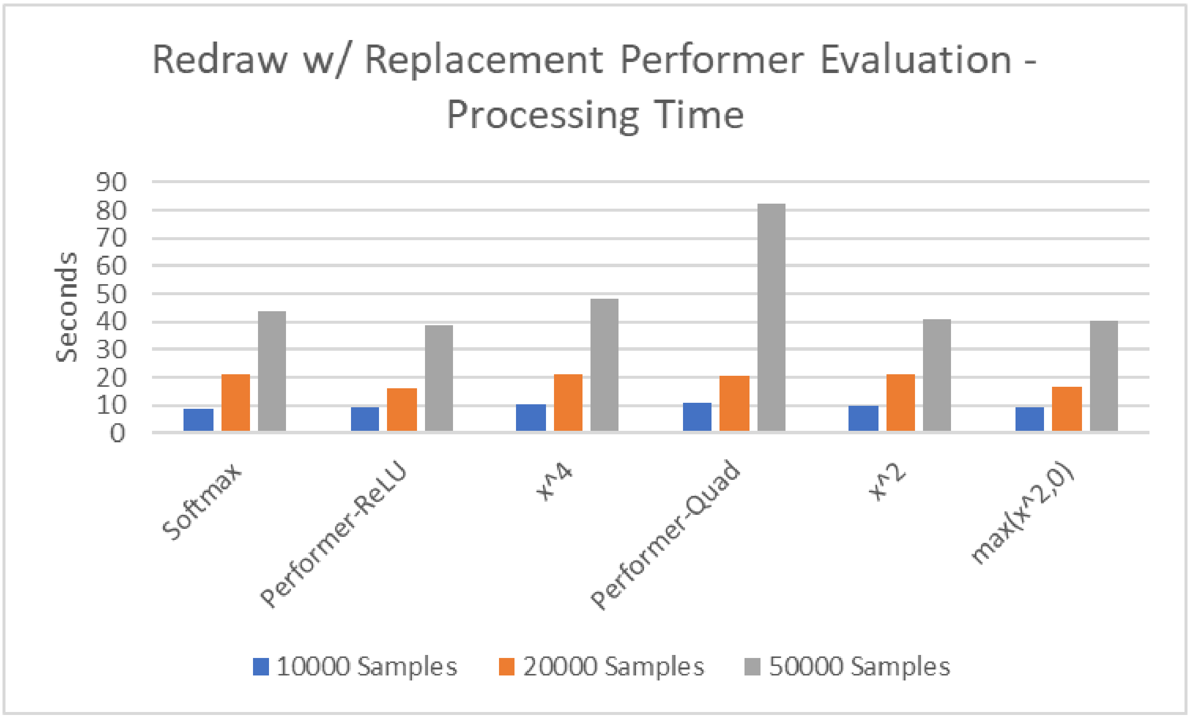

Figure 6: Redraw with Replacement Performer Processing Times.

Conclusion

Using the performer attention kernels overall resulted in better model performance than a regular vision transformer with the exception of the performer-ReLU activation. Sampling from the images with and without replacement did not yield any noticeable difference across all trials. As the number of samples increased, model performance was better and processing time was longer, which makes sense since more computations have been performed. Further work in this project could be done in testing more activation functions with and without the Performer architecture. Also, as mentioned above, one random seed can be used across all trials to reduce the limited variability between trials. Finally, testing could be extended to other image datasets to assess performance further.

References

Choromanski, Krzysztof, et al. “Rethinking Attention with Performers.” *ArXiv.org*, 19 Nov. 2022, https://arxiv.org/abs/2009.14794.

Grother, Patrick J. “NIST Special Database 19 Handprinted Forms and Characters Database.” 16 Mar. 1995, https://www.nist.gov/system/files/documents/srd/nistsd19.pdf.