Recasting Self-Attention with Holographic Reduced Representations

Mohammad Mahmudul Alam
University of Maryland, Baltimore County
m256@umbc.edu

Edward Raff
Booz Allen Hamilton
Raff_Edward@bah.com

Tim Oates
University of Maryland, Baltimore County
oates@umbc.edu

James Holt
Laboratory for Physical Science
holt@lps.umd.edu

Abstract

Self-Attention has become fundamentally a new approach to set and sequence modeling, particularly within transformer-style architectures. Given a sequence of T items the standard self-attention has $O(T^2)$ memory and compute needs, leading to many recent works building approximations to self-attention with reduced computational or memory complexity. In this work, we instead re-cast self-attention using the neuro-symbolic approach of Holographic Reduced Representations (HRR). In doing so we perform the same logical strategy of the standard self-attention. Implemented as a "Hrrformer" we obtain several benefits including faster compute ($O(T \log T)$ time complexity), less memory-use per layer ($O(T)$ space complexity), convergence in 10× fewer epochs, near state-of-the-art accuracy, and we are able to learn with just a single layer. Combined, these benefits make our Hrrformer up to 370× faster to train on the Long Range Arena benchmark.

Keywords: sequence classification, self-attention, transformers

ACM Reference Format:
Mohammad Mahmudul Alam, Edward Raff, Tim Oates, and James Holt. 2022. Recasting Self-Attention with Holographic Reduced Representations. In *Proceedings of 8TH SIGKDD INTERNATIONAL WORKSHOP ON MINING AND LEARNING FROM TIME SERIES – DEEP FORECASTING: MODELS, INTERPRETABILITY, AND APPLICATIONS (MiLeTS 2022).* ACM, New York, NY, USA, 9 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Self-attention has become a key component of the development of Transformer style architectures, and their recent successes in machine translation, large language modeling, and computer vision applications. The fundamental construction of self-attention includes a triplet of "queries, keys, and values", where the response is a weighted average over the values based on the query-key interactions. This results in a quadratic memory and computational complexity, that has inhibited the use of Transformers to those without significant GPU infrastructure and prevented applications to longer sequences. Ever since a myriad of approaches has been proposed to approximate the self-attention mechanism, with the vast majority trading some amount of accuracy for speed or memory use. The "market" of self-attention strategies currently offers various trade-offs in the total package of speed, memory use, and accuracy.

In this work, we take a different approach to improving the self-attention mechanism. Rather than approximate the existing self-attention approach, we re-design the same logical strategy: return a weighted sum of values based on the best match pairing of a set of queries against the keys for each value. To do this we use a neuro-symbolic approach of Holographic Reduced Representations (HRR) [36]. The HRR allows us to model the same logic in a mechanism that is differentiable, thus allowing integration into a larger neural network. We term the Transformer built from our new mechanism a Hrrformer (pronounced \her-for-mer\).

We test our method using the Long Range Arena (LRA) to compare with numerous prior results, as well as a real-world task in Malware detection. These results show several benefits to the Hrrformer: it is near state-of-the-art in terms of accuracy, and one of only two methods to improve upon the original Transformer for *all* tasks in the LRA. The Hrrformer sets a new state-of-the-art for speed and memory use, processing 37× more samples/second and 76.79% less memory than the best prior arts for each respective metric. The Hrrformer converges in 10× fewer epochs and is effective with just a single layer. Combined this makes the Hrrformer up to 370× times faster to train. On our malware classification task, we find that the relative accuracies of

Transformer models change from the LRA benchmark, but that our Hrrformer still obtains the best accuracy and scales the best with sequence length up to $T = 8192$.

The remainder of our manuscript is organized as follows. Work related to our own, as well as adjacent techniques beyond our study's scope, are reviewed in section 2. The recasting of attention in our Hrrformer is a simple procedure demonstrated in section 3, which redefines the *Attention* function using HRR, and multi-headed self-attention then continues as normal. We then demonstrate these benefits in section 4, showing Hrrformer is consistently one of the best methods in accuracy and considerably faster thanks to reduced memory usage, the number of layers, and epochs needed to converge. In section 5 we conclude our work.

2 Related Works

Since the introduction of the Self-Attention mechanism and the transformer architecture, considerable research has occurred to mitigate its computational burdens. Though not explicit in much of the current literature, many of these approaches resemble strategies for improving Support Vector Machines that have a similar complexity. This includes projection [20] to a lower dimension ([59]), finding/creating sparse structure in the correlations [58] (by [6, 10, 24, 54, 63]), using randomized features [43, 49] (by [11]), factorized or budgeted representations [47, 60] (by [31, 62]), and creating simplified linear approximations [21, 61] (by [22]. Other more differentiated approaches include the hierarchical decomposition of the correlations (by [65]), and approaches that replace self-attention entirely with alternative "mixing" strategies [27, 53]. To the best of our knowledge, ours is the first work that attempts to re-create the same logic of self-attention with alternative constructs (the HRR), without applying approximations to the attention mechanism or using alternative mixing.

Among these prior methods, we note that F-Net [27] is the most related as both F-Net and HRR rely upon the Fast Fourier Transform (FFT) as a fundamental building block. While F-Net does not approximate self-attention so much as replace it with an alternative "mixing" procedure, we include it due to its relevance in using the FFT. Our results will show significant improvement over F-Net, highlighting the value of a neuro-symbolic approach to reconstructing the same logic as opposed to using the FFT as a generic differentiable mixing strategy.

The HRR has seen successful use in cognitive science research [5, 7, 8, 12, 19, 48, 52], but comparatively little work in modern deep learning. The symbolic properties have been previously used in knowledge graphs [35] multi-label classification [13], and privacy [3], but never previously for sequential modeling. An older alternative to the HRR, the Tensor Product Representation (TPR) [50] has been used to endow Recurrent Neural Networks with enhanced functionality [18, 46]. Compared to these prior works, we are re-casting the same logic into HRRs, rather than augmenting the logic. In addition, the TPR's complexity is at least quadratic, making it a poor choice for tackling the scaling problems of self-attention.

Other recent approaches to sequential modeling like IGLOO [51] and State Space Models[14–17] are highly promising. We consider these, along with RNNs, beyond the scope of our work. Our goal is to explore the value of re-casting self-attention within the neuro-symbolic framework of HRR. As such other sequence modeling approaches are out of scope, and we focus purely on other self-attention mechanisms.

The need for both less memory and extension to very long sequences is also prevalent in Malware detection. Processing malware from raw bytes has been found to be one of the most robust feature types in the face of common malware obfuscations [2], but simple n-gram based features have been maligned for being unable to learn complex sequential information when executable can be tens of kilobytes on the small side and hundreds of megabytes on the larger side [1, 23, 25, 39, 64]. Given a maximum $T = 200M$ is realistic, many strategies to handle such sequence lengths have developed. This includes attempts to create "images" from malware have been attempted [30, 34], using compression algorithms as a similarity metric [9, 28, 33, 41, 42, 45, 57], and attempts to scale 1D-convolutional networks over raw bytes [26, 38, 40].

We will use the Ember[4] dataset for malware detection as a real-world test of our new self-attention. We are aware of no current work that has successfully applied Transformers to the raw bytes task, and we do not yet get Transformers to process an entire executable. Still, we find the experiment informative to the robustness of Transformer architectures to other domains, where results change for current state-of-the-art methods but our Hrrformer maintains its high performance. The task is also interesting as prior work has identified cyber security data as behaving in intrinsically different manners to current broad understanding [44] which is consistent with prior observations about convolutional networks [38].

3 Attention with Holographic Reduced Representations

The HRR operation allows assigning abstract concepts to arbitrary numerical vectors, and perform *binding* ($\circledast$) and *unbinding* operations on those concepts via the vectors. One could bind "red" and "cat" to obtain a "red cat". The vectors can also be added, so "red" $\circledast$ "cat" + "yellow" $\circledast$ "dog" represents a "red cat and yellow dog". An inverse operator $\dagger$ is used to perform unbinding. One can then query a bound representation, asking "what was red?" by unbinding "red cat and yellow dog" $\circledast$ "red"† to get a vector $\approx$ "cat". To perform this symbolic manipulation the binding operation is defined as Equation 1

$$\mathcal{B} = \mathbf{x} \circledast \mathbf{y} = \mathcal{F}^{-1}(\mathcal{F}(\mathbf{x}_i) \odot \mathcal{F}(\mathbf{y}_i)) \tag{1}$$

where $\mathcal{F}$ denotes the FFT and $\odot$ an element-wise multiplication. The inversion is defined as $\mathbf{y}^\dagger = \mathcal{F}^{-1}\left(\frac{1}{\mathcal{F}(\mathbf{y})}\right)$. Combined Plate showed that the response $\mathcal{B} \circledast \mathbf{y}^\dagger$ should be ≈ 1 if the vector $\mathbf{y} \in \mathcal{B}$, and ≈ 0 if not present.

An attention function can be represented using query $\mathbf{Q}$, key $\mathbf{K}$, and value $\mathbf{V}$ vectors where the final output is computed as the weighted sum of the values. A query vector can be mapped to a set of linked key-value pairs to retrieve the value vector associated with the query. The concept of *binding* and *unbinding* operations of HRR is applied to link the key-value pair, i.e., bound term, and retrieve the value vector associated with the query from the bound term.

Given a sequence of length T having an embedding size of H for $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{T \times H}$, the key and value vectors are paired together into a bound vector $\boldsymbol{\beta}$ using the binding operation.

$$\boldsymbol{\beta} = \mathbf{K} \circledast \mathbf{V} \in \mathbb{R}^{T \times H} \tag{2}$$

Next, to make a composite representation of the bound terms, all the elements of the sequence are added together into a single synthetic vector represented by $\boldsymbol{\beta}^+$.

$$\boldsymbol{\beta}^+ = \sum_{i=1}^{T} \boldsymbol{\beta}_i \in \mathbb{R}^{1 \times H} \tag{3}$$

Later, the query vector is applied to retrieve the corresponding value vector $\hat{\mathbf{V}}$ linked with the key associated with that query using the unbinding operation.

$$\hat{\mathbf{V}} = \boldsymbol{\beta}^+ \circledast \mathbf{Q}^\dagger \in \mathbb{R}^{T \times H} \tag{4}$$

The retrieved value vector $\hat{\mathbf{V}}$ is extracted from a synthetic vector storing all the information of a sequence using the query vector $\mathbf{Q}$. Therefore, a similarity score between the original value vector $\mathbf{V}$ and the retrieved value vector $\hat{\mathbf{V}}$, will have the information of which element of the sequence to prioritize. As a result, cosine similarity is calculated between them which is scaled using the softmax activation to get the weight $\mathbf{W}$ values of a sequence.

$$\mathbf{W} = \text{softmax}\left(\frac{\sum_{i=1}^{H} \mathbf{V}_i \cdot \hat{\mathbf{V}}_i}{\|\mathbf{V}\|_2 \cdot \|\hat{\mathbf{V}}\|_2}\right) \in \mathbb{R}^{T \times 1} \tag{5}$$

Finally, the weight vector $\mathbf{W}$ is multiplied element-wise with the original value vector to compute the final attention given in Equation 6.

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathbf{W} \cdot \mathbf{V} \in \mathbb{R}^{T \times H} \tag{6}$$

Moreover, we expand our attention mechanism to multiple heads. Instead of performing single attention, we split the embedding size H of the query, key, and value into h heads each having an embedding of size $H' = H/h$. The attention is computed in parallel in each head and then merged into single attention which is projected to get the final output.

The time complexity of the binding/unbinding operation is $O(T \log T)$. Therefore, the time complexity of the Hrrformer attention per layer is log-linear $O(T \log T \cdot H)$ whereas the space complexity is linear $O(T)$.

This simple approach allows us to have fully replicated the same logical goals and construction of the attention mechanism first proposed by [56]. This neuro-symbolic reconstruction yields several benefits, as we will demonstrate in the next section of our experiments. By simply replacing the self-attention in a standard Transformer with our HRR-based self-attention gives the "Hrrformer" that we will use to judge the utility of this new derivation.

4 Experiments and Results

The proposed Hrrformer is designed as an inexpensive alternative to the self-attention models for longer sequences. Experiments are performed to validate the effectiveness of the method in terms of time and space complexity in known benchmarks.

Our first results will use the Long Range Arena (LRA) [55] which has become a standard for evaluations in this space. The primary value of these results is to compare our Hrrformer with numerous prior works, establishing the broad benefits of faster time per batch, convergence in 10× fewer epochs, requiring only a single layer, and competitive overall accuracy.

Our second result is running many of the current popular and state-of-the-art xformers on the real-world classification task of the Ember malware detection dataset [4]. This provides an example where the need to handle ever longer sequences still exists and demonstrates that Hrrformer is one of the fastest and most accurate options on a problem with complex dynamics that exceed that of the LRA. In doing so we also show that current "SotA" methods like Luna-256 do not generalize as well to new problem spaces, as our Hrrformer does.

4.1 Long Range Arena

The Long Range Arena (LRA) [55] benchmark comprises 6 diverse tasks covering image, text, math, language, and spatial modeling under long context scenarios ranging from $1K$ to $16K$. **ListOps** – task inspects the capability of modeling hierarchically structured data in a longer sequence context with mathematical operators `MAX`, `MEAN`, `MEDIAN`, and `SUM MOD` enclosed by delimiters. This is a ten-way classification problem with a maximum sequence length of $2K$. **Text** – is a byte/character level classification task that uses IMDB movie review [32] dataset. Character-level language modeling makes the models reason with compositional unsegmented data to provide a meaningful result. This is a binary classification task with a maximum sequence length of $4K$. **Retrieval** – evaluates the model's ability to encode and compress useful for matching and retrieval by modeling

Table 1. Accuracy results of Hrrformer on Long Range Arena (LRA) benchmark. Our Multi-layer results use the same layer count (3-6) per task as prior methods. Even using just one layer Hrrformer is highly competitive, and the only method besides Luna to be a Pareto improvement over the original Transformer. Our method is further advantaged in that it requires 10× fewer epochs to reach competitive accuracies.

Model	ListOps (2k)	Text (4k)	Retrieval (4k)	Image (1k)	Path (1k)	Path-X (16k)	Avg	Epochs
Transformer [56]	36.37	64.27	57.46	42.44	71.40	FAIL	54.39	200
Local Attention [55]	15.82	52.98	53.39	41.46	66.63	FAIL	46.06	200
Linear Transformer [22]	16.13	65.90	53.09	42.34	75.30	FAIL	50.55	200
Reformer [24]	37.27	56.10	53.40	38.07	68.50	FAIL	50.67	200
Sparse Transformer [10]	17.07	63.58	59.59	44.24	71.71	FAIL	51.24	200
Sinkhorn Transformer [54]	33.67	61.20	53.83	41.23	67.45	FAIL	51.29	200
Linformer [59]	35.70	53.94	52.27	38.56	76.34	FAIL	51.36	200
Performer [11]	18.01	65.40	53.82	42.77	77.05	FAIL	51.41	200
Synthesizer [53]	36.99	61.68	54.67	41.61	69.45	FAIL	52.88	200
Longformer [6]	35.63	62.85	56.89	42.22	69.71	FAIL	53.46	200
BigBird [63]	36.05	64.02	59.29	40.83	74.87	FAIL	55.01	200
F-Net [27]	35.33	65.11	59.61	38.67	*77.78*	FAIL	54.42	200
Nystromformer [62]	37.15	65.52	**79.56**	41.58	70.94	FAIL	58.95	200
Luna-256 [31]	37.98	65.78	79.56	47.86	**78.55**	FAIL	**61.95**	200
H-Transformer-1D [65]	**49.53**	**78.69**	63.99	46.05	68.78	FAIL	61.41	200
Hrrformer Single-layer	38.86	66.49	75.13	47.90	72.79	FAIL	60.23	**20**
Hrrformer Multi-layer	38.24	65.90	75.83	**48.41**	73.17	FAIL	60.31	**20**

similarity score between two documents. For this task, the ACL Anthology Network [37] dataset is used in a character level setup. This task has a maximum sequence length of $8K$ and this is a binary classification task. **Image** – is an image classification task of 10 classes that uses grayscale CIFAR-10 dataset in a sequence of length $32\times32 = 1024$. This task allows assessing the model's ability to process discrete symbols. **Pathfinder** – task evaluates the model's performance over long-range spatial dependency. This is a binary classification task that classifies whether two circles are connected by a line which is introduced in [29]. To make the task even more challenging, the images contain distractor paths. The images have dimension 32×32 which is reshaped into 1024.**Path-X** is extremely difficult version of pathfinder task which contains images of dimension $128 \times 128 = 16384$ with additional distractor paths. This task determines whether the same algorithmic proficiency scales up to the longer sequences.

In Hrrformer, we use the same number of parameters as mentioned in the LRA benchmark [55] across the tasks. Global average pooling is applied to the output of the encoder sequences and subsequently back to back dense layers are used with ReLU activation to get the final logits output. During training, the softmax cross-entropy loss function is optimized using the Adam optimizer. We use the exponential decay learning rate with the initial value of 10^{-3}, the final value of 10^{-5}, and the decay rate of 0.9. For all the tasks, Hrrformer is trained for a total of 20 epochs both in the case of single- and multi-layer which is 10× less than the previous works. The results in terms of accuracy in all the tasks of the LRA benchmark are presented in Table 1. [1]

Ours is one of only two methods that improve accuracy upon the Transformer and consistently acquired higher performance in all the tasks. All the models listed in Table 1 uses 3 to 6 layers of encoder. We show the performance for both single and multiple layers. In 3 of the 5 tasks (ListOps, Text, Image), Hrrformer achieves the second-best results using only 1 layer of the encoder. For the Image classification task, it achieves the best results of 48.41% accuracy using 3 layers of the encoder. Moreover, Hrrformer requires 10× fewer epochs than others to produce comparable and better results. Overall, the multi-layered Hrrformer produces the second-best result of 60.31% in the benchmark.

The ability to learn with a single layer aids in both throughput and memory use. The result is surprising, and in visualizing the weight vector $\mathbf{W}$ of Equation 5 we can confirm that

[1]We note that the Pathfinder task as originally reported by [55] uses a "hard" version of the task, but the code provided defaults to an "easy" version. Most papers do not make clear which version of the task is evaluated, and the F-Net authors indicated in correspondence the "easy" version was used. Luna-256 used the hard version, and other authors have not yet reached back to us. On the easy version, Hrrformer gets 80.81% in a single-layer and 80.77% in the multi-layer, but we report the hard version in our table and assume others are using the hard version.

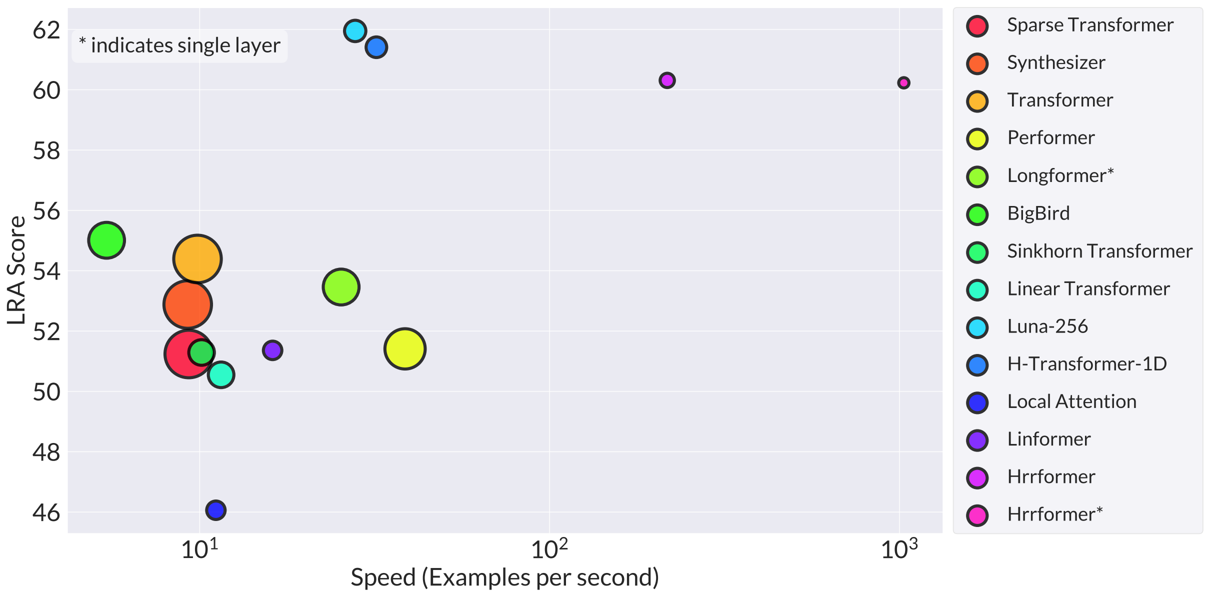

Figure 1. Performance (y-axis), Speed (x-axis, log-scale) of different xformers, and memory footprint on GPU are illustrated by the size of the circles. Hrrformer is in the top-right of the graph, with the smallest circle size, indicating it is the fastest and most memory efficient for training (this does *not* factor in convergence speed).

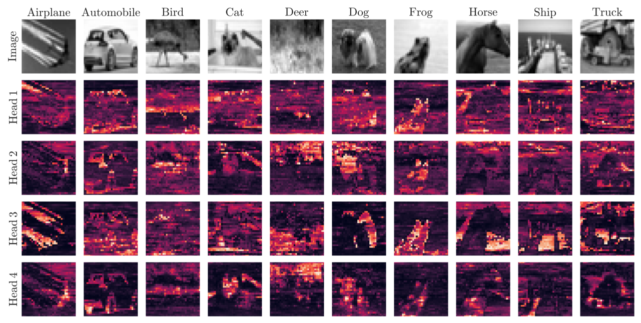

Figure 2. Visualization of weight vector $\mathbf{W} \in \mathbb{R}^{1024\times1}$ reshaped to 32×32, the shape of the original image of the CIFAR-10 dataset used in the LRA Image classification task. A single layer Hrrformer is able to learn the 2D structure from the 1D sequence of the image. This is particularly noticeable in the Airplane, dog, Frog, and Horse images. Note context sensitive Head activation can be observed comparing Head 3 for dog vs Frog, where activation occurs for different pixel intensities indicating the model is not naively activating for simple color intensity.

a single layer is sufficient to learn the structure. We show this for the Image task of single-layer Hrrformer in Figure 2. Here, the weight vector $\mathbf{W} \in \mathbb{R}^{1024\times1}$ is reshaped to 32×32, the shape of the original grayscale images of the CIFAR-10 dataset for visualization. From the figure, it is clear that the Hrrformer is learning to identify the 2D structure from the 1D sequence of the Image classification task.

Hrrformer's benefits go beyond accuracy and convergence speed: it is fast and consumes the least amount of memory on GPU. Figure 1 compares all the self-attention models in terms of LRA score, speed (training examples per second), and memory footprint (size of the circle). LRA score is the mean accuracy of all the tasks in the LRA benchmark. Speed and memory footprint is calculated on the byte-level text classification task. To measure these results, a single NVIDIA TESLA PH402 32GB GPU is utilized with a fixed batch size of 16 and a maximum sequence length of 1000. For all the models 6 layers of the encoder are used. Both single- and multi-layered Hrrformer are 37× and 7.8× faster than the Luna-256 [31] which has achieved the highest accuracy in the LRA benchmark. Hrrformer also consumes the least amount of memory, taking 76.79% and 55.24% less memory compared to Luna-256 in the case of single and multi-layered Hrrformer, respectively.

Hrrformer also reduces the amount of overfitting between train and test performance. We compare the training and test accuracy, and amount of overfitting of the Image classification task to the other self-attention models presented in LRA benchmark [55] and for which data is available[2]. Table 2 exhibits that the Hrrformer acquires the best results on the test set with an 8.45% train/test gap. The learning curves of all the task is also presented in Figure 4 demonstrating the lower overfitting nature of the Hrrformer across the tasks.

[2]We do not have the compute resources to run the other xformers on the LRA ourselves, in part due to the higher memory use that exceeds our infrastructure.

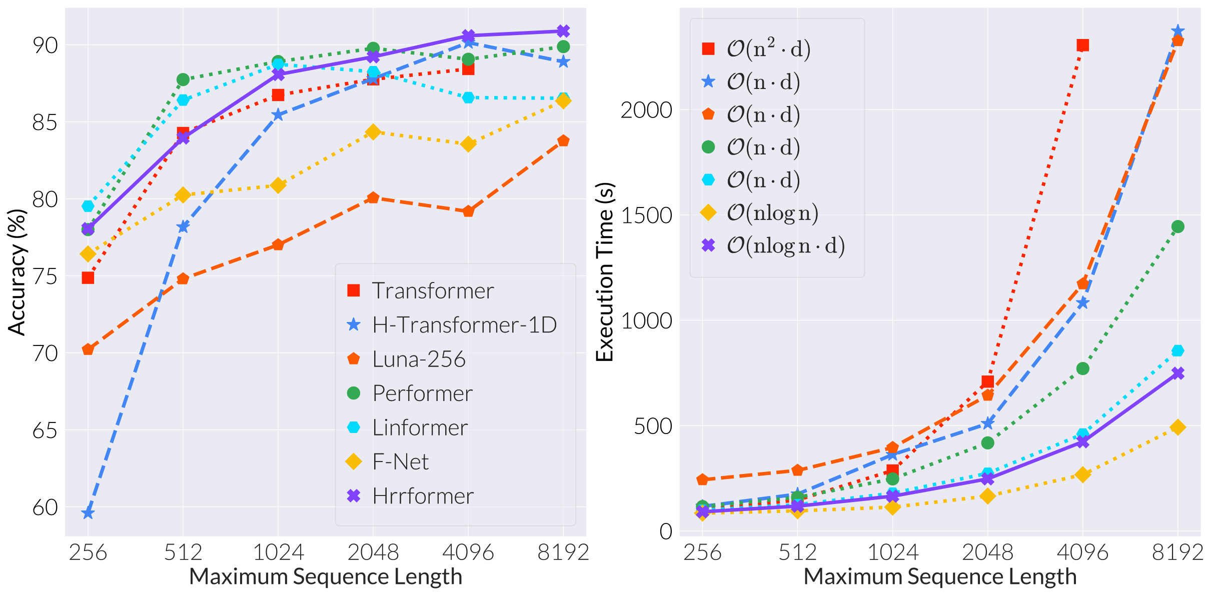

Figure 3. Comparison of Hrrformer with other self-attention models in EMBER malware classification dataset. Hrrformer is presented in a *solid* line and achieves the best accuracy, and second fastest run-time. The two prior best models according the the Long Range Arena H-Transformer-1D and Luna-256 are in the *dashed* lines, and do not perform as well as the LRA would have indicated in speed or accuracy. The rest of the models are in the *dotted* line. This shows the Hrrformer is one of the best options in benchmarks and real-world tasks.

Table 2. Train and test accuracy of different self-attention models on the Image classification task. Among all the models, Hrrformer achieves the best test accuracy with the least amount of overfitting (lower is better).

Model	Train Acc. (%)	Test Acc. (%)	Overfit (%)
Transformer	69.45	42.44	27.01
Local Attention	63.19	41.46	21.73
Sparse Transformer	66.74	44.24	22.50
Longformer	71.65	42.22	29.43
Linformer	97.23	38.56	58.67
Reformer	68.45	38.07	30.38
Sinkhorn Transformer	69.21	41.23	27.98
Synthesizer	**97.31**	41.61	55.70
BigBird	71.49	40.83	30.66
Linear Transformer	65.61	42.34	23.27
Performer	73.90	42.77	31.13
Hrrformer	56.86	**48.41**	**8.45**

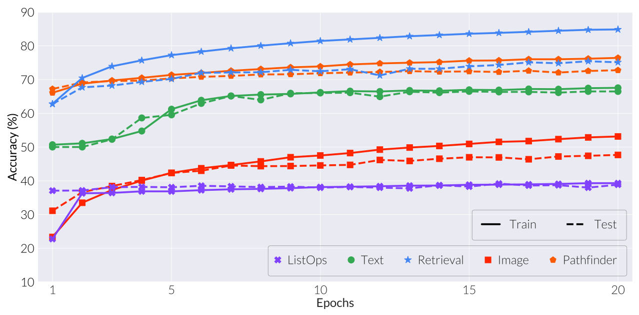

Figure 4. Learning curves of Hrrformer in the LRA tasks, training performance are solid lines and test is dashed. Where prior works required 200 epochs of training, we can see that 20 epochs is sufficient for our Hrrformer. In the Pathfinder, Retrieval, and ListOps task the single-epoch performance of our Hrrformer is still highly competitive.

4.2 EMBER

EMBER is a benchmark dataset for the malware classification task [4]. The benchmark contains 600K labeled training samples (300K malicious, 300K benign) and 200K labeled test samples (100K malicious, 100K benign). The maximum sequence length of this dataset is over 100M which is not feasible for any of the self-attention models to train with. We experiment with relatively shorter sequence lengths starting from $T = 256$ and doubling up to $T = 8192$ by truncating the bytes until this maximum length is reached.

In this benchmark, Hrrformer is compared with Transformer [56], H-Transformer-1D [65], Luna-256 [31], Performer [11], Linformer [59], and F-Net [27] self-attention models. In all those models, 8 heads of a single encoder with 256 embedding size and 512 hidden size of the feed-forward network are used. Since this is a binary classification task, the encoder output is mapped into 2 logits output using back-to-back dense layers with ReLU activation. During training, the softmax cross-entropy loss function is optimized.

For sequence length 256, the batch size is set to be 256. In the experiment, as the sequence length doubles, we halved the batch size to fit the data and the model to the memory which can be expressed as $\max(2^{16-\log_2 T},\ 1)$. This is done to push other models to the maximum possible length, and keep the batch size consistent between experiments. Even after exponentially decaying batch size, we could not fit the transformer model to the memory for the sequence length 8196. The dropout rate is chosen to be 0.1, the learning rate is 10^{-3} with an exponential decay rate of 0.85. Each of the models is trained for a total of 10 epochs in 16 NVIDIA TESLA PH402 32GB GPUs.

Figure 3 shows the classification accuracy and the execution time of each of the methods for incremental sequence length. As the sequence length increases, Hrrformer outperforms the rest of the models achieving the highest 90.89% accuracy for maximum sequence length 8192. In terms of execution time, F-Net is the only model that is faster than ours, however, the accuracy of F-Net is 4.53% lagging behind us. The detailed numeric results are presented in **??**. The execution time for linear time complexity methods seems quadratic in the figure, this is due to the exponential decay of the batch size with the increase of sequence length.

Of significant importance to our results is that Luna-256 performs considerably worse than all other options, compared to its top accuracy in the LRA. We hypothesize that the Ember task requires more complex reasoning and feature extraction over time and because Luna performs aggressive compression and approximation of the time component of the model it suffers in terms of accuracy. Our Hrrformer on the other hand has consistent behavior across Ember and the LRA: high accuracy, able to handle longer sequences, and convergence in few epochs, a requirement for working on this dataset which is 1 TB in size and is otherwise prohibitive in its scale.

5 Conclusion

In this paper, we have presented Hrrformer a neuro-symbolic reconstruction of self-attention. The proposed method is faster in compute and consumes less memory per layer. We have tested Hrrformer on known LRA and EMBER benchmarks. In the LRA benchmark, Hrrformer has achieved the near state-of-the-art accuracy of 60.23% using a single layer of an encoder. In terms of speed, it is 37× and 7.8× faster than the current state-of-the-art in the case of single and multiple layers, respectively. Additionally, it takes 76.79% and 55.24% less memory on GPU compared to Luna-256 for single and multiple layers of Hrrformer. Besides, it converges 10× faster than other self-attention models. In the EMBER malware classification dataset, Hrrformer has attained the highest 90.89% accuracy for a maximum sequence length of 8192 with a significantly faster processing rate. In conclusion, Hrrformer is $\approx 370\times$ faster to train and a single layer of the encoder is sufficient to learn the structure of the input.

References

[1] Tony Abou-Assaleh, Nick Cercone, Vlado Keselj, and Ray Sweidan. 2004. N-gram-based detection of new malicious code. In *Proceedings of the 28th Annual International Computer Software and Applications Conference, 2004. COMPSAC 2004.*, Vol. 2. IEEE, 41–42. https://doi.org/10.1109/CMPSAC.2004.1342667

[2] Hojjat Aghakhani, Fabio Gritti, Francesco Mecca, Martina Lindorfer, Stefano Ortolani, Davide Balzarotti, Giovanni Vigna, and Christopher Kruegel. 2020. When Malware is Packin' Heat; Limits of Machine Learning Classifiers Based on Static Analysis Features. In *Proceedings 2020 Network and Distributed System Security Symposium*. Internet Society, Reston, VA. https://doi.org/10.14722/ndss.2020.24310

[3] Mohammad Mahmudul Alam, Edward Raff, Tim Oates, and James Holt. 2022. Deploying Convolutional Networks on Untrusted Platforms Using 2D Holographic Reduced Representations. In *International Conference on Machine Learning*. http://arxiv.org/abs/2206.05893

[4] Hyrum S Anderson and Phil Roth. 2018. Ember: an open dataset for training static pe malware machine learning models. *arXiv preprint arXiv:1804.04637* (2018).

[5] Trevor Bekolay, James Bergstra, Eric Hunsberger, Travis DeWolf, Terrence Stewart, Daniel Rasmussen, Xuan Choo, Aaron Voelker, and Chris Eliasmith. 2014. Nengo: a Python tool for building large-scale functional brain models. *Frontiers in Neuroinformatics* 7 (2014), 48. https://doi.org/10.3389/fninf.2013.00048

[6] Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150* (2020).

[7] Peter Blouw and Chris Eliasmith. 2013. A Neurally Plausible Encoding of Word Order Information into a Semantic Vector Space. *35th Annual Conference of the Cognitive Science Society* 35 (2013), 1905–1910.

[8] Peter Blouw, Eugene Solodkin, Paul Thagard, and Chris Eliasmith. 2016. Concepts as Semantic Pointers: A Framework and Computational Model. *Cognitive Science* 40, 5 (7 2016), 1128–1162. https://doi.org/10.1111/cogs.12265

[9] Rebecca Schuller Borbely. 2015. On normalized compression distance and large malware. *Journal of Computer Virology and Hacking Techniques* (2015), 1–8. https://doi.org/10.1007/s11416-015-0260-0

[10] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509* (2019).

[11] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. 2020. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794* (2020).

[12] C. Eliasmith, T. C. Stewart, X. Choo, T. Bekolay, T. DeWolf, Y. Tang, and D. Rasmussen. 2012. A Large-Scale Model of the Functioning Brain. *Science* 338, 6111 (11 2012), 1202–1205. https://doi.org/10.1126/science.1225266

[13] Ashwinkumar Ganesan, Hang Gao, Sunil Gandhi, Edward Raff, Tim Oates, James Holt, and Mark McLean. 2021. Learning with Holographic Reduced Representations. In *Advances in Neural Information Processing Systems*. http://arxiv.org/abs/2109.02157

[14] Karan Goel, Albert Gu, Chris Donahue, and Christopher Ré. 2022. It's Raw! Audio Generation with State-Space Models. *arXiv* (2022), 1–23. http://arxiv.org/abs/2202.09729

[15] Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Ré. 2020. HiPPO: Recurrent memory with optimal polynomial projections.

Advances in Neural Information Processing Systems (2020).
[16] Albert Gu, Karan Goel, and Christopher Ré. 2022. Efficiently Modeling Long Sequences with Structured State Spaces. In *ICLR*. http://arxiv.org/abs/2111.00396
[17] Albert Gu, Isys Johnson, Karan Goel, Khaled Saab, Tri Dao, Atri Rudra, and Christopher Ré. 2021. Combining Recurrent, Convolutional, and Continuous-time Models with Linear State-Space Layers. In *NeurIPS*. http://arxiv.org/abs/2110.13985
[18] Qiuyuan Huang, Paul Smolensky, Xiaodong He, Li Deng, and Dapeng Wu. 2018. Tensor Product Generation Networks for Deep NLP Modeling. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Association for Computational Linguistics, New Orleans, Louisiana, 1263–1273. https://doi.org/10.18653/v1/N18-1114
[19] Michael N. Jones and Douglas J.K. Mewhort. 2007. Representing word meaning and order information in a composite holographic lexicon. *Psychological Review* 114, 1 (2007), 1–37. https://doi.org/10.1037/0033-295X.114.1.1
[20] Ata Kaban. 2015. Improved Bounds on the Dot Product under Random Projection and Random Sign Projection. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '15)*. Association for Computing Machinery, New York, NY, USA, 487–496. https://doi.org/10.1145/2783258.2783364
[21] Alex Kantchelian, Michael Carl Tschantz, Ling Huang, Peter L Bartlett, Anthony D Joseph, and J D Tygar. 2014. Large-margin Convex Polytope Machine. In *Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS'14)*. MIT Press, Cambridge, MA, USA, 3248–3256. http://dl.acm.org/citation.cfm?id=2969033.2969189
[22] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning*. PMLR, 5156–5165.
[23] Jeffrey O Kephart, Gregory B Sorkin, William C Arnold, David M Chess, Gerald J Tesauro, and Steve R White. 1995. Biologically Inspired Defenses Against Computer Viruses. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 1 (IJCAI'95)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 985–996. http://dl.acm.org/citation.cfm?id=1625855.1625983
[24] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451* (2020).
[25] J Zico Kolter and Marcus A Maloof. 2006. Learning to Detect and Classify Malicious Executables in the Wild. *Journal of Machine Learning Research* 7 (12 2006), 2721–2744. http://dl.acm.org/citation.cfm?id=1248547.1248646
[26] Marek Krčál, Ondřej Švec, Martin Bálek, and Otakar Jašek. 2018. Deep Convolutional Malware Classifiers Can Learn from Raw Executables and Labels Only. In *ICLR Workshop*.
[27] James Lee-Thorp, Joshua Ainslie, Ilya Eckstein, and Santiago Ontanon. 2021. FNet: Mixing Tokens with Fourier Transforms. *arXiv* (2021). http://arxiv.org/abs/2105.03824
[28] Ming Li, Xin Chen, Xin Li, Bin Ma, and Paul M.B. Vitanyi. 2004. The Similarity Metric. *IEEE Transactions on Information Theory* 50, 12 (2004), 3250–3264. https://doi.org/10.1109/TIT.2004.838101
[29] Drew Linsley, Junkyung Kim, Vijay Veerabadran, Charles Windolf, and Thomas Serre. 2018. Learning long-range spatial dependencies with horizontal gated recurrent units. *Advances in neural information processing systems* 31 (2018).
[30] Liu Liu and Baosheng Wang. 2016. Malware classification using grayscale images and ensemble learning. In *2016 3rd International Conference on Systems and Informatics (ICSAI)*. IEEE, 1018–1022. https://doi.org/10.1109/ICSAI.2016.7811100
[31] Xuezhe Ma, Xiang Kong, Sinong Wang, Chunting Zhou, Jonathan May, Hao Ma, and Luke Zettlemoyer. 2021. Luna: Linear Unified Nested Attention. In *NeurIPS*. http://arxiv.org/abs/2106.01540
[32] Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*. 142–150.
[33] Héctor D Menéndez, Sukriti Bhattacharya, David Clark, and Earl T Barr. 2019. The arms race: Adversarial search defeats entropy used to detect malware. *Expert Systems with Applications* 118 (2019), 246–260. https://doi.org/10.1016/j.eswa.2018.10.011
[34] L Nataraj, S Karthikeyan, G Jacob, and B S Manjunath. 2011. Malware Images: Visualization and Automatic Classification. In *Proceedings of the 8th International Symposium on Visualization for Cyber Security (VizSec '11)*. ACM, New York, NY, USA, 4:1–4:7. https://doi.org/10.1145/2016904.2016908
[35] Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. 2016. Holographic Embeddings of Knowledge Graphs. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI'16)*. AAAI Press, 1955–1961.
[36] Tony A Plate. 1995. Holographic reduced representations. *IEEE Transactions on Neural networks* 6, 3 (1995), 623–641.
[37] Dragomir R Radev, Pradeep Muthukrishnan, Vahed Qazvinian, and Amjad Abu-Jbara. 2013. The ACL anthology network corpus. *Language Resources and Evaluation* 47, 4 (2013), 919–944.
[38] Edward Raff, Jon Barker, Jared Sylvester, Robert Brandon, Bryan Catanzaro, and Charles Nicholas. 2018. Malware Detection by Eating a Whole EXE. In *AAAI Workshop on Artificial Intelligence for Cyber Security*. http://arxiv.org/abs/1710.09435
[39] Edward Raff, William Fleming, Richard Zak, Hyrum Anderson, Bill Finlayson, Charles K. Nicholas, Mark Mclean, William Fleming, Charles K. Nicholas, Richard Zak, and Mark Mclean. 2019. KiloGrams: Very Large N-Grams for Malware Classification. In *Proceedings of KDD 2019 Workshop on Learning and Mining for Cybersecurity (LEMINCS'19)*. https://arxiv.org/abs/1908.00200
[40] Edward Raff, William Fleshman, Richard Zak, Hyrum S. Anderson, Bobby Filar, and Mark McLean. 2021. Classifying Sequences of Extreme Length with Constant Memory Applied to Malware Detection. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence*. http://arxiv.org/abs/2012.09390
[41] Edward Raff and Charles Nicholas. 2017. An Alternative to NCD for Large Sequences, Lempel-Ziv Jaccard Distance. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '17*. ACM Press, New York, New York, USA, 1007–1015. https://doi.org/10.1145/3097983.3098111
[42] Edward Raff, Charles Nicholas, and Mark McLean. 2020. A New Burrows Wheeler Transform Markov Distance. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence*. 5444–5453. https://doi.org/10.1609/aaai.v34i04.5994
[43] Ali Rahimi and Ben Recht. 2007. Random Features for Large-Scale Kernel Machines. In *Neural Information Processing Systems*. http://seattle.intel-research.net/pubs/rahimi-recht-random-features.pdf
[44] Ethan M. Rudd and Ahmed Abdallah. 2020. Training Transformers for Information Security Tasks: A Case Study on Malicious URL Prediction. *arXiv* (2020). https://doi.org/10.48550/arXiv.2011.03040
[45] João S. Resende, Rolando Martins, and Luís Antunes. 2019. A Survey on Using Kolmogorov Complexity in Cybersecurity. *Entropy* 21, 12 (12 2019), 1196. https://doi.org/10.3390/e21121196
[46] Imanol Schlag and Jürgen Schmidhuber. 2018. Learning to Reason with Third Order Tensor Products. In *Advances in Neural Information Processing Systems*, S Bengio, H Wallach, H Larochelle, K Grauman, N Cesa-Bianchi, and R Garnett (Eds.), Vol. 31. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2018/file/a274315e1abede44d63005826249d1df-Paper.pdf
[47] Si Si, Cho-Jui Hsieh, and Inderjit S Dhillon. 2016. Computationally Efficient Nystrom Approximation using Fast Transforms. In *International*

Conference on Machine Learning (ICML).

[48] Ray Singh and Chris Eliasmith. 2006. Higher-Dimensional Neurons Explain the Tuning and Dynamics of Working Memory Cells. *Journal of Neuroscience* 26, 14 (2006), 3667–3678. https://doi.org/10.1523/JNEUROSCI.4864-05.2006

[49] Aman Sinha and John C Duchi. 2016. Learning Kernels with Random Features. In *Advances In Neural Information Processing Systems 29*, D D Lee, U V Luxburg, I Guyon, and R Garnett (Eds.). Curran Associates, Inc., 1298–1306. http://papers.nips.cc/paper/6180-learning-kernels-with-random-features.pdf

[50] Paul Smolensky. 1990. Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artificial Intelligence* 46, 1 (1990), 159–216. https://doi.org/10.1016/0004-3702(90)90007-M

[51] Vsevolod Sourkov. 2018. IGLOO: Slicing the Features Space to Represent Long Sequences. *arXiv* (2018). http://arxiv.org/abs/1807.03402

[52] Terrence C. Stewart and Chris Eliasmith. 2014. Large-scale synthesis of functional spiking neural circuits. *Proc. IEEE* 102, 5 (2014), 881–898. https://doi.org/10.1109/JPROC.2014.2306061

[53] Y Tay, D Bahri, D Metzler, D Juan, Z Zhao, and C Zheng. 2020. Synthesizer: Rethinking self-attention in transformer models. *arXiv preprint arXiv:2005.00743* (2020).

[54] Yi Tay, Dara Bahri, Liu Yang, Donald Metzler, and Da-Cheng Juan. 2020. Sparse sinkhorn attention. In *International Conference on Machine Learning*. PMLR, 9438–9447.

[55] Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. 2020. Long range arena: A benchmark for efficient transformers. *arXiv preprint arXiv:2011.04006* (2020).

[56] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[57] Andrew Walenstein and Arun Lakhotia. 2007. The Software Similarity Problem in Malware Analysis. *Duplication, Redundancy, and Similarity in Software* (2007). http://drops.dagstuhl.de/opus/volltexte/2007/964

[58] Jie Wang, Peter Wonka, and Jieping Ye. 2014. Scaling SVM and Least Absolute Deviations via Exact Data Reduction. *JMLR W&CP* 32, 1 (2014), 523–531.

[59] Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. 2020. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768* (2020).

[60] Zhuang Wang, Koby Crammer, and S Vucetic. 2010. Multi-class pegasos on a budget. In *27th International Conference on Machine Learning*. 1143–1150. http://www.ist.temple.edu/~vucetic/documents/wang10icml.pdf

[61] Zhuang Wang, Nemanja Djuric, Koby Crammer, and Slobodan Vucetic. 2011. Trading representability for scalability Adaptive Multi-Hyperplane Machine for nonlinear Classification. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '11*. ACM Press, New York, New York, USA, 24. https://doi.org/10.1145/2020408.2020420

[62] Yunyang Xiong, Zhanpeng Zeng, Rudrasis Chakraborty, Mingxing Tan, Glenn Fung, Yin Li, and Vikas Singh. 2021. Nystr\"omformer: A Nystr\"om-Based Algorithm for Approximating Self-Attention. In *AAAI*. http://arxiv.org/abs/2102.03902

[63] Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems* 33 (2020), 17283–17297.

[64] Richard Zak, Edward Raff, and Charles Nicholas. 2017. What can N-grams learn for malware detection?. In *2017 12th International Conference on Malicious and Unwanted Software (MALWARE)*. IEEE, 109–118. https://doi.org/10.1109/MALWARE.2017.8323963

[65] Zhenhai Zhu and Radu Soricut. 2021. H-Transformer-1D: Fast One-Dimensional Hierarchical Attention for Sequences. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Online, 3801–3815. https://doi.org/10.18653/v1/2021.acl-long.294