A Proof of Theorem 1

Here we provide a detailed proof to show that SBM-Transformer is a universal approximator of arbitrary sequence-to-sequence functions. Note that a trivial solution is to use a dense mask $\boldsymbol{M}$ equal to the all-one matrix with rank 1, in which case SBM-Transformer becomes equivalent to the full attention Transformer [7] that is already known to achieve universal approximability [13]. Instead, we show that there also exists a solution with $\mathcal{O}(n)$ connections, leveraging previous analyses under sparse attention by Yun et al. (2020) [14] and Zaheer et al. (2020) [15].

For theoretical analysis, we consider a variant of SBM-Transformer that manually adds self-loops in the bipartite graph such that $\boldsymbol{M}_{ii} = 1$ for all i. While adding in self-loops help towards analyzing expressibility, we find that it does not help empirically, and hence omit the modification in the our main method during experimentation. A comparison on performance on the LRA benchmark can be found below in Appendix C.

Here we restate the necessary conditions from [14]. Let $\mathcal{A}_i^l \subseteq [n]$ denote the sparsity pattern of i-th token in the l attention pattern: $j \in \mathcal{A}_i^l$ if query i attends to key j in the l-th pattern. Then, the main theorem of Yun et al.,(2020) [14] states that as long as the set of p sparsity patterns $\{\mathcal{A}_i^l\}_{l=1}^p$ and the probability mapping ρ (*e.g.*, softmax) of the sparse Transformer model satisfy the two assumptions below, then model achieves universal approximability with finite number of layers.

Assumption 1. *The sparsity patterns $\{\mathcal{A}_i^l\}$ satisfy the following:*

1. *For all $i \in [n]$ and $l \in [p]$, we have $i \in \mathcal{A}_i^l$*
2. *There exists a permutation $\gamma : [n] \to [n]$ such that, for all $i \in [n-1]$, $\gamma(i) \in \cup_{l=1}^p \mathcal{A}_{\gamma(i+1)}^l$.*
3. *There exists a finite $s \in \mathbb{N}$ such that $s = \min\{u \mid \mathcal{S}_i^u = [n] \text{ for all } i \in [n]\}$ where $\mathcal{S}_i^u$ is defined recursively by $\mathcal{S}_i^1 \coloneqq \mathcal{A}_k^1$ and $\mathcal{S}_i^t \coloneqq \bigcup_{j \in \mathcal{A}_i^{(t-1) \bmod p+1}} \mathcal{S}_j^{t-1}$.*

Assumption 2. *For any $\zeta > 0$ and $\eta \in (0, 1]$, $\exists t > 0$ such that, for any column input $\boldsymbol{v}$ satisfying $v_{j^*} - \max_{j \neq j^*} v_j \geq \zeta$ (where $j^* = \arg\max_j v_j$), we have $\rho[t\boldsymbol{v}]_{j^*} \geq 1 - \eta$ and $\sum_{j \neq j^*} \rho[t\boldsymbol{v}]_j \leq \eta$*

When viewing each attention pattern $\mathcal{A}^l$ as a directed graph $G^l = (V, E^l)$ with node set $V \coloneqq [n]$ and edge set $E^l \coloneqq \{(j, i) | j \in \mathcal{A}_i^l \; \forall i, j\}$, each item in Assumption 1 can be equivalently written as

Condition 1: For all directed graphs G^l, each node has a self-loop.

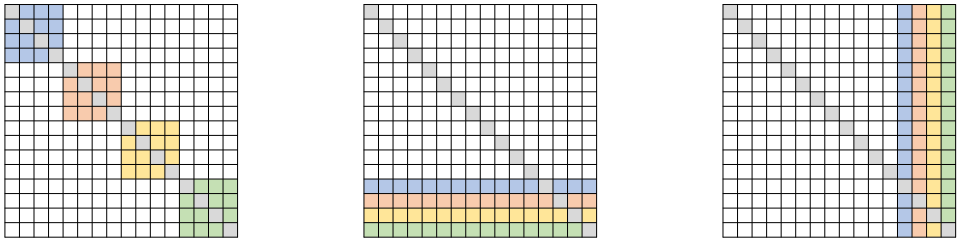
Condition 2: The aggregation of all p graphs $G^* \coloneqq (V, \cup_{l=1}^p E^l)$ has a Hamiltonian path that spans all n nodes.

Condition 3: In a finite aggregation of s graphs $G^{*s} \coloneqq (V, \cup_{l=1}^s E^l)$, there exists a path between all possible pairs of nodes.

Because we use the same softmax probability mapping, which is already proven to satisfy Assumption 2 in [14], we are left to show that there exists a parameterization of SBM-Transformer such that the expected attention mask patterns together satisfy the three conditions above. To do so, we first show that a simple random ER-graph [1] can be expected to have at least one Hamiltonian cycle with expected number of edges linear in the sequence length.

Lemma 1. *Assume a directed Erdős-Rényi random graph $G(n, p)$ where each directed edge exists with probability p. Then, for any number of nodes n, there exists a probability p such that the expected number of edges is $\mathcal{O}(n)$ and the expected number of Hamiltonian cycles in $G(n, p)$ is greater than or equal to 1.*

Proof. We start the proof by formulating the expected number of Hamiltonian cycles in $G(n, p)$. Assuming directed edges, there exist $(n-1)!$ permutations, each of which represent different possible Hamiltonian cycles. Say we have $(n-1)!$ random variables $\{X_i\}_i^{(n-1)!}$ where each X_i equals 1 when the corresponding Hamiltonian cycle exists in G, 0 otherwise. By linearity of expectation, the expected number of Hamiltonian cycles equals $\sum_{i=1}^{(n-1)!} \mathbb{E}[X_i]$. Then, note the probability of $X_i = 1$ equals p^n for all i since we require n directed edges to exist for each cycle. Therefore, the total expected number of Hamiltonian cycles equals $\sum_{i=1}^{(n-1)!} \mathbb{E}[X_i] = p^n(n-1)!$.



(a) $\mathcal{A}^1$ (b) $\mathcal{A}^2$ (c) $\mathcal{A}^3$

Figure 1: Three sparsity patterns with $n = 16$ and $k = 4$. Grey-colored blocks on the diagonal indicate manually added self-loops. Any other color indicates a cluster.

Next, we show that $\sum_{i=1}^{(n-1)!} \mathbb{E}[X_i] \geq 1$ if $p = f(n)$ where $f(n) = \mathcal{O}(\frac{1}{n})$. Starting from $\sum_{i=1}^{(n-1)!} \mathbb{E}[X_i] = p^n(n-1)!$, using the inequality $n! \geq (n/e)^n$ leads to

$$p^n(n-1)! = \frac{p^n}{n} n! \geq \frac{p^n}{n} \left(\frac{n}{e}\right)^n$$

Then, setting the RHS equal to 1 leads to

$$\frac{p^n}{n} \left(\frac{n}{e}\right)^n = 1 \Leftrightarrow n \ln p + n \ln \frac{n}{e} = \ln n \Leftrightarrow \ln p = \ln \frac{e}{n} + \ln n^{\frac{1}{n}} \Leftrightarrow p = \frac{e}{n} n^{\frac{1}{n}}$$

For large n, $\frac{1}{n}$ dominates $n^{\frac{1}{n}}$ and thus, the expected number of Hamiltonian cycle is larger than or equal to 1 with expected number of edges $n^2 p = \mathcal{O}(n)$ □

Lemma 2. *There exists a parameterization of SBM-Transformer such that the sparsity patterns induced by the expected attention masks satisfy Assumption 1.*

Proof. Here we show that a finite number of attention patterns each representable by the SBM given some number of clusters k achieves the three conditions from Assumption 1. Here we use $p = 3$ attention patterns together (shown in Figure 1):

$$\mathcal{A}^1_i = \{i\} \cup \left\{ j : \left\lfloor \frac{ik}{n} \right\rfloor = \left\lfloor \frac{jk}{n} \right\rfloor \; \forall j \in [n] \right\} \text{ for all } i \in [n]$$

$$\mathcal{A}^2_i = \{i\} \cup \{n-k+1, \ldots, n-1, n\} \text{ for all } i \in [n]$$

$$\mathcal{A}^3_i = \begin{cases} \{i\} & \text{if } i \leq n-k \\ [n] & \text{if } i > n-k \end{cases}$$

Intuitively speaking, $\mathcal{A}^1$ clusters all tokens into non-overlapping k clusters, each with size $\frac{n}{k}$, and connects tokens together if they are within the same cluster. The other two patterns $\mathcal{A}^2$ and $\mathcal{A}^3$ adds k global relay tokens for each cluster with edges going from and to all n nodes, respectively. Note that all three patterns are easily representable from separate SBMs.

Then, we can show that these three patterns form directed graphs that together satisfy the three required conditions. Condition 1 is easily satisfied due to the manually added self-loops in all patterns. Condition 3 is also satisfied with $s = 3$ as we have k global relay tokens in both directions ($\mathcal{A}^2$ and $\mathcal{A}^3$), connecting all pairs of tokens indirectly or directly. Lastly, Condition 2 can be satisfied by leveraging Lemma 1 and the global k relay tokens: Lemma 1 states that each subgraph induced by each individual cluster in $\mathcal{A}^1$ has at least one Hamiltonian cycle with $\mathcal{O}(n)$ number of edges in expectation. Then, a global Hamiltonian path can in G^* can be constructed as follows:

- Traverse through the first induced subgraph using its Hamiltonian cycle in $\mathcal{A}^1$, but without going back to the starting node.
- Move to the $n-k+1$ global relay token via the edge in $\mathcal{A}^2$, then move to any node in the second induced subgraph from node $n-k+1$ via an edge in $\mathcal{A}^3$.
- Traverse through the Hamiltonian cycle in the second induced subgraph, and repeat.

This way, we can construct a global Hamiltonian path that visits all n nodes, and all three conditions are met with $\mathcal{O}(kn)$ number of edges in expectation. □

Combining Lemma 2 together with Theorem 1 of Yun et al. (2020) [14] proves our main theorem below which states that SBM-Transformer is a universal approximator in expectation.

Theorem 1. *Let $f \in \mathcal{F}$ be class of continuous sequence-to-sequence functions. Let $\mathcal{T}_{SBM}^{h,m,r}$ denote the class of SBM-Transformers with h attention heads, m head dimension, and r dimensions in hidden layers. Then for any $\epsilon > 0$ and $1 \leq p < \infty$, there exists a function $g \in \mathcal{T}_{SBM}^{h,m,r}$ such that*

$$\int_{\mathbb{D}} \|f(\boldsymbol{X}) - \mathbb{E}[g(\boldsymbol{X})]\|_p^p d\boldsymbol{X} \leq \epsilon$$

B Asymptotic Cost Analysis

Table 1 shows the asymptotic computational cost and memory footprint of each step an attention head takes in SBM-Transformer given a single input. Assuming the number of clusters is significantly smaller than the sequence length, we find that both time and memory cost is mostly dominated by the computation of $\hat{\boldsymbol{Q}}$ and $\hat{\boldsymbol{K}}$ when the sampled graph is sparse (*i.e.*, $m = \mathcal{O}(n)$).

Computation	Time	Memory
Inputs $\boldsymbol{Q}$, $\boldsymbol{K}$, $\boldsymbol{V}$, and $\boldsymbol{C}$	-	$\mathcal{O}(nd + kd)$
Node assignments $\hat{\boldsymbol{Q}}$ and $\hat{\boldsymbol{K}}$	$\mathcal{O}(nd^2 + nkd)$	$\mathcal{O}(nd + kd + nk)$
Inter-cluster probabilities $\hat{\boldsymbol{S}}$	$\mathcal{O}(k^2 d)$	$\mathcal{O}(k^2)$
Sampling from `fastRG` [6]	$\mathcal{O}(m + n)$[1]	$\mathcal{O}(m + nk + k^2)$
Run GAT [8] with edge-softmax	$\mathcal{O}(md)$	$\mathcal{O}(m + nd)$[2]
Total	$\mathcal{O}(md + nd^2 + nkd + k^2 d)$	$\mathcal{O}(m + nd + nk + kd + k^2)$

Table 1: Asymptotic costs of individual steps within the attention module of SBM-Transformer. The sequence length, number of edges, number of clusters, and head dimension are denoted as n, m, k, and d, respectively.

A comparison of the overall cost of SBM-Transformer with those of other baselines is shown in Table 2. While its complexities most resemble those of Nyströmformer [12] when the sampled graphs are sparse, the cost of SBM-Transformer can exceed those of full-attention when the graph is dense, due to the additional computation in the $\text{MLP}_{d \to d}$ used to infer node-to-cluster memberships.

Model	Time	Memory
Full-attention [7]	$\mathcal{O}(n^2 d)$	$\mathcal{O}(n^2 + nd)$
Linearized [4]	$\mathcal{O}(nd^2)$	$\mathcal{O}(nd + d^2)$
Reformer [5]	$\mathcal{O}(nd + nk(4n/c)^2)$	$\mathcal{O}(nd + nk(4n/c)^2)$
Performer [2]	$\mathcal{O}(nkd + kd^2)$	$\mathcal{O}(nk + nd)$
Linformer [11]	$\mathcal{O}(nkd + nk)$	$\mathcal{O}(nk + nd)$
Nyströmformer [12]	$\mathcal{O}(nkd + nk^2 + k^3)$	$\mathcal{O}(nk + nd + kd + k^2)$
SBM-Transformer (ours)	$\mathcal{O}(md + nd^2 + nkd + k^2 d)$	$\mathcal{O}(m + nd + nk + kd + k^2)$

Table 2: Asymptotic computational costs of different attention mechanisms. The k term denotes different parameters for each model: number of clusters for SBM-Transformer, number of hashing rounds for Reformer [5], number of random features for Performer [2], the projection rank for Linformer [11], and the number of landmarks for Nyströmformer [12]. The additional c term in Reformer [5] indicates the number of hashing chunks, set to $c = \mathcal{O}(\frac{1}{n})$ as default.

[1] Walker's Alias Method [9] used to sample nodes in `fastRG` requires $\mathcal{O}(m + n \log n)$ operations, but the $\log n$ dependency is not visible in general. More information can be found in [6]

[2] We leverage highly optimized Generalized Sampled-Dense-Dense Matrix Multiplication (GSDDMM) operators provided by the Deep Graph Library [10] that avoids the $\mathcal{O}(md)$ memory overhead.

C Experiments

For reproducibility, we list the model and training hyperparameter settings used for each task in Table 3. Note that for SBM-Transformer, we initialize the cluster-embeddings $\boldsymbol{C}$ using the kaiming normal distribution [3], which results in an initial attention density of approximately 25%. Tables 4 and 5 provide the full LRA benchmark results with standard deviations in test-time accuracy and sparsity. As mentioned in the main paper, we find that manually fixing the self-loops in the sampled graphs slightly deteriorates performance, while it helps in proving theoretical expressibility.

Parameter	SYNTHETIC	LISTOPS	TEXT	RETRIEVAL	IMAGE	PATHFINDER	BERT	GLUE
# of layers	1	2	2	2	2	2	4	4
# of heads	1	2	2	2	2	2	8	8
Embedding dim.	32	64	64	64	64	64	512	512
Hidden dim.	32	128	128	128	128	128	2048	2048
Head dim.	32	32	32	32	32	32	64	64
Sequence len.	256	2048	3072	4096	1024	1024	512	512
Dropout	0.0	0.1	0.1	0.1	0.1	0.1	0.1	0.1
Attn. dropout	0.0	0.1	0.1	0.1	0.1	0.1	0.1	0.1
Pooling mode	N/A	MEAN	MEAN	MEAN	MEAN	MEAN	N/A	MEAN
# of classes	2	10	2	2	10	2	50265	2 or 3
Batch size	256	128	128	32	1024	1024	256	32
Learning rate	1e-3	5e-4	5e-4	1e-4	5e-4	5e-4	1e-4	3e-5
# of training epochs	2000	5000	20000	30000	35000	62400	50	5

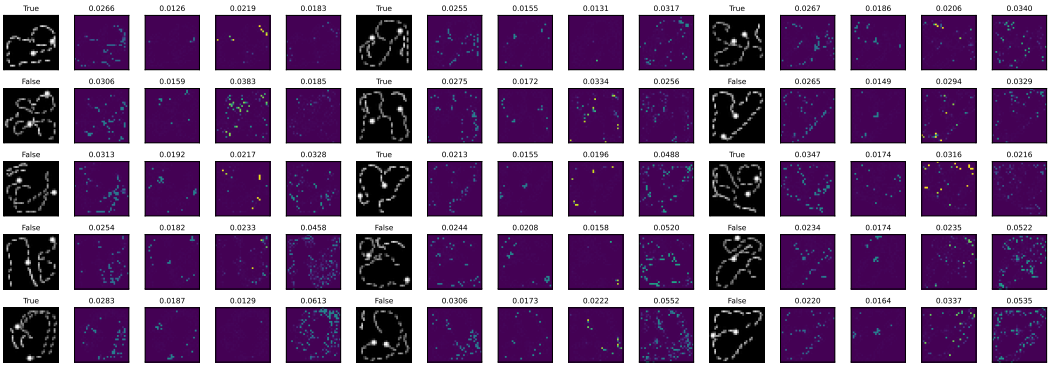
Table 3: Hyperparameter settings used synthetic, LRA, and GLUE experiments. For methods other than full attention [7], we use 128 clusters for SBM-Transformer, 2 hashing rounds for Reformer [5], 256 landmarks for Nyströmformer [12], and 256 dimensions for Linformer [11] and Performer [2].

Model	LISTOPS(2K)	TEXT(3K)	RETRIEVAL(4K)	IMAGE(1K)	PATHFINDER(1K)	Avg.
Full-attention [7]	37.22±0.52	64.93±0.46	79.55±1.22	40.38±0.76	74.26±0.57	59.27±0.44
Linearized [4]	37.46±0.57	64.90±0.49	**81.10±0.16**	38.48±0.57	74.61±1.26	59.31±0.15
Reformer [5]	22.92±0.41	64.70±0.12	77.25±0.15	**43.65±0.16**	70.28±1.45	55.76±0.29
Performer [2]	18.25±0.12	65.00±0.50	79.01±1.66	39.80±0.46	70.79±1.26	54.57±0.55
Linformer [11]	**38.44±0.14**	56.28±1.06	78.09±0.12	39.53±0.57	67.62±0.65	55.99±0.14
Nyströmformer [12]	37.22±0.51	<u>65.46±0.40</u>	79.35±0.40	<u>43.07±0.42</u>	71.97±1.30	<u>59.41±0.12</u>
SBM-Transformer ($+\boldsymbol{I}$)	<u>37.60±0.38</u> (<u>24.64±2.49%</u>)	64.09±1.39 (25.64±0.64%)	79.74±0.27 (24.26±5.21%)	40.64±0.72 (24.54±3.98%)	<u>74.93±0.32</u> (<u>23.84±3.59%</u>)	59.40±0.20
SBM-Transformer ($+\mathbf{0}$)	37.45±0.44 (20.09±15.71%)	**65.79±0.27** (**26.10±0.01%**)	<u>80.00±0.21</u> (<u>29.46±3.84%</u>)	41.31±0.35 (20.49±11.43%)	**75.12±0.49** (**18.56±0.52%**)	**59.93±0.35**

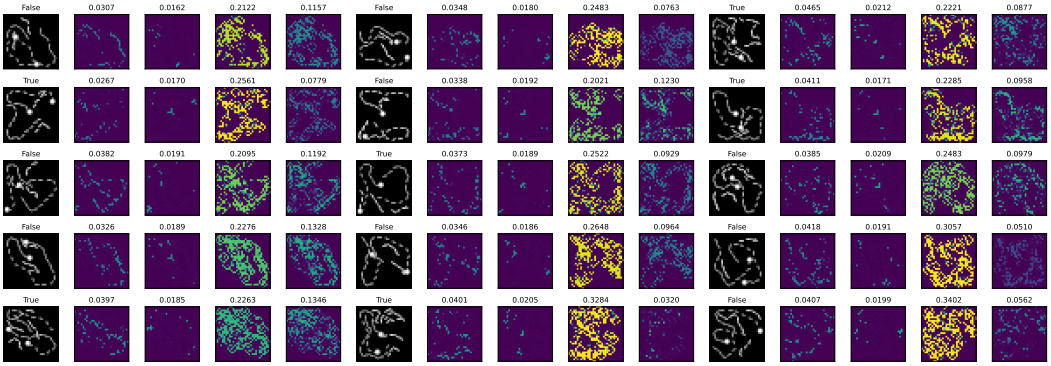
Table 4: LRA benchmark results. Bold and underlined results indicate best and 2nd best test accuracy for each task, respectively. Numbers enclosed in parentheses for SBM-Transformer indicate the density of graphs sampled during test time averaged across all attention heads. For the SBM-Transformer models, $(+\boldsymbol{I})$ indicates that self-loops are manually fixed while $(+\mathbf{0})$ indicates model without the modification.

λ	LISTOPS(2K)	TEXT(3K)	RETRIEVAL(4K)	IMAGE(1K)	PATHFINDER(1K)	Avg.
0	37.45±0.44 (20.09±15.71%)	**65.79±0.27** (**26.10±0.01%**)	<u>80.00±0.21</u> (<u>29.46±3.84%</u>)	41.31±0.35 (20.49±11.43%)	<u>75.12±0.49</u> (<u>18.56±0.52%</u>)	<u>59.93±0.35</u>
10^{-4}	37.76±0.60 (10.48±7.58%)	65.48±0.86 (26.26±0.53%)	79.93±0.16 (24.62±3.19%)	<u>41.35±0.35</u> (<u>10.70±8.49%</u>)	**75.46±0.46** (**5.16±1.17%**)	**60.00±0.36**
10^{-3}	**38.23±0.63** (**10.46±7.26%**)	65.18±0.46 (26.03±0.06%)	80.00±0.99 (21.70±2.68%)	41.17±0.53 (24.60±8.61%)	74.49±0.74 (3.82±0.52%)	59.81±0.48
10^{-2}	<u>38.20±0.29</u> (<u>2.95±0.88%</u>)	<u>65.59±0.24</u> (<u>22.43±1.73%</u>)	**80.44±1.24** (**6.99±2.28%**)	**42.20±0.64** (**3.95±0.68%**)	72.79±0.80 (3.76±0.27%)	59.84±0.42
10^{-1}	37.76±0.83 (1.15±0.15%)	64.48±0.58 (10.62±2.74%)	79.46±0.47 (2.49±0.58%)	41.35±0.40 (1.33±0.37%)	73.79±0.07 (2.61±0.22%)	59.37±0.37

Table 5: LRA benchmark results of SBM-Transformer with increasing density regularization weight λ. Applying a density regularizer helps in encouraging sparser attention patterns which induce less computational cost, while retaining competitive performance.



(a) Examples with low attention density



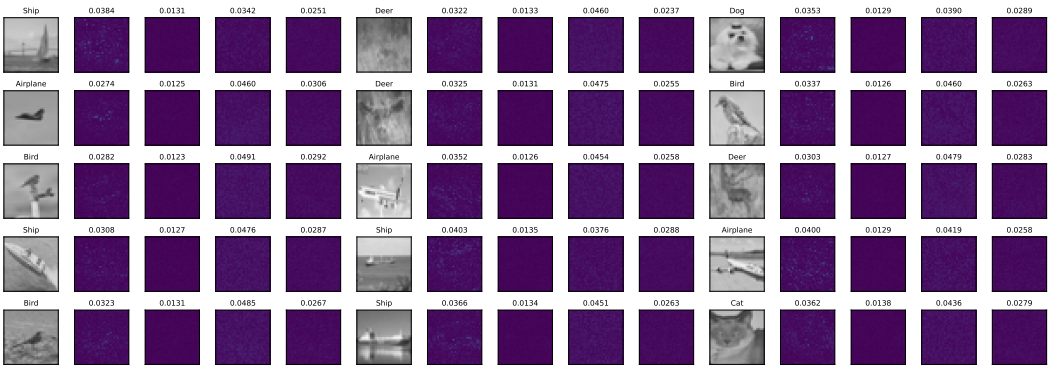
(b) Examples with high attention density

Figure 2: Attention density plots within individual attention heads given inputs from the LRA PATHFINDER test set. All examples shown are from a subset of the test set that the model has predicted correctly. For each set of 5 images, the leftmost image shows the original input image of which the title shows the ground-truth label. To its right are attention density plots from two heads of the first layer followed by those from two heads of the second layer. Above each plot is the actual numeric attention density between 0 and 1. The color in each pixel indicates how many other pixels attend to that particular pixel (a color closer to bright yellow indicates more attention).

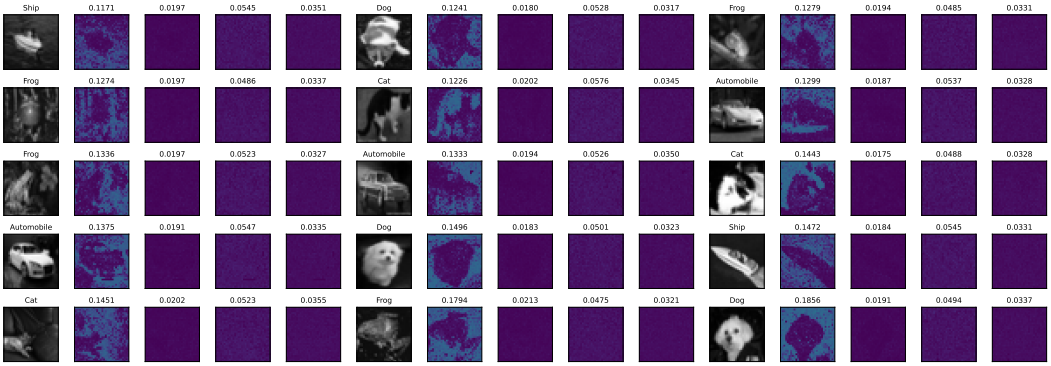
Lastly, we qualitatively analyze which inputs lead to sparse or dense attention in SBM-Transformer. For easy visualization of attention densities, we choose two image-based tasks in LRA, PATHFINDER and IMAGE. We pick two model checkpoints that performed best on each of the two tasks under graph density regularization, one trained with $\lambda = 10^{-4}$ for PATHFINDER and another trained with $\lambda = 10^{-2}$ for IMAGE, and run predictions on the respective test sets. Figures 2 and 3 show the head-wise attention densities per input at different levels.

In Figure 2, the second layer shows large variance in attention density across different PATHFINDER inputs, while the first layer remains sparse overall. With some exceptions, we find that the attention density of this layer is somewhat correlated with the difficulty of each input. Figure 2a shows visually easy inputs with near-perpendicular intersections or no intersection at all, allowing correct predictions with less than 5% average attention density. On the other hand, Figure 2b shows examples with harder difficulty, due to having more lines and convoluted intersections. We can see that the model uses much denser attention in such cases, and thus conjecture that the model is adaptively choosing to look at more pixel-to-pixel interactions in response to the complexity of the input.

Figure 3 also shows a clear distinction between images that induce different levels of attention density. Under regularization, the first layer of SBM-Transformer focuses attention onto dark areas in the image as shown in Figure 3b, using the contrast in the image for better prediction. When the image has high overall intensity as in Figure 3a, however, the model uses less than 3% attention on average, focusing most of the prediction onto the skip-connections, FFNs, and a small number of pixel-to-pixel interactions. Considering that this model achieves a competitive 42.20% accuracy, this shows that SBM-Transformer can well balance the tradeoff between computational cost vs. performance, further supporting the power of our adaptively sparse attention module.



(a) Examples with low attention density



(b) Examples with high attention density

Figure 3: Similar visualization as Figure 2 for the LRA IMAGE test set.

References

[1] B. Bollobás. Random graphs. In *Modern graph theory*, pages 215–252. Springer, 1998.

[2] K. Choromanski, V. Likhosherstov, D. Dohan, X. Song, A. Gane, T. Sarlós, P. Hawkins, J. Davis, A. Mohiuddin, L. Kaiser, D. Belanger, L. J. Colwell, and A. Weller. Rethinking attention with performers. *CoRR*, abs/2009.14794, 2020.

[3] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *CoRR*, abs/1502.01852, 2015.

[4] A. Katharopoulos, A. Vyas, N. Pappas, and F. Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. *CoRR*, abs/2006.16236, 2020.

[5] N. Kitaev, L. Kaiser, and A. Levskaya. Reformer: The efficient transformer. *CoRR*, abs/2001.04451, 2020.

[6] K. Rohe, J. Tao, X. Han, and N. Binkiewicz. A note on quickly sampling a sparse matrix with low rank expectation. *The Journal of Machine Learning Research*, 19(1):3040–3052, 2018.

[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[8] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph Attention Networks. *International Conference on Learning Representations*, 2018. accepted as poster.

[9] A. J. Walker. An efficient method for generating discrete random variables with general distributions. *ACM Trans. Math. Softw.*, 3(3):253–256, sep 1977.

[10] M. Wang, D. Zheng, Z. Ye, Q. Gan, M. Li, X. Song, J. Zhou, C. Ma, L. Yu, Y. Gai, T. Xiao, T. He, G. Karypis, J. Li, and Z. Zhang. Deep graph library: A graph-centric, highly-performant package for graph neural networks. *arXiv preprint arXiv:1909.01315*, 2019.

[11] S. Wang, B. Z. Li, M. Khabsa, H. Fang, and H. Ma. Linformer: Self-attention with linear complexity. *CoRR*, abs/2006.04768, 2020.

[12] Y. Xiong, Z. Zeng, R. Chakraborty, M. Tan, G. Fung, Y. Li, and V. Singh. Nyströmformer: A nyström-based algorithm for approximating self-attention. *CoRR*, abs/2102.03902, 2021.

[13] C. Yun, S. Bhojanapalli, A. S. Rawat, S. J. Reddi, and S. Kumar. Are transformers universal approximators of sequence-to-sequence functions? *CoRR*, abs/1912.10077, 2019.

[14] C. Yun, Y. Chang, S. Bhojanapalli, A. S. Rawat, S. J. Reddi, and S. Kumar. O(n) connections are expressive enough: Universal approximability of sparse transformers. *CoRR*, abs/2006.04862, 2020.

[15] M. Zaheer, G. Guruganesh, A. Dubey, J. Ainslie, C. Alberti, S. Ontañón, P. Pham, A. Ravula, Q. Wang, L. Yang, and A. Ahmed. Big bird: Transformers for longer sequences. *CoRR*, abs/2007.14062, 2020.