What Dense Graph Do You Need for Self-attention?

Xipeng Qiu

Fudan University

xpqiu@fudan.edu.cn

29 Apr 2022

The Vanilla Transformer

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

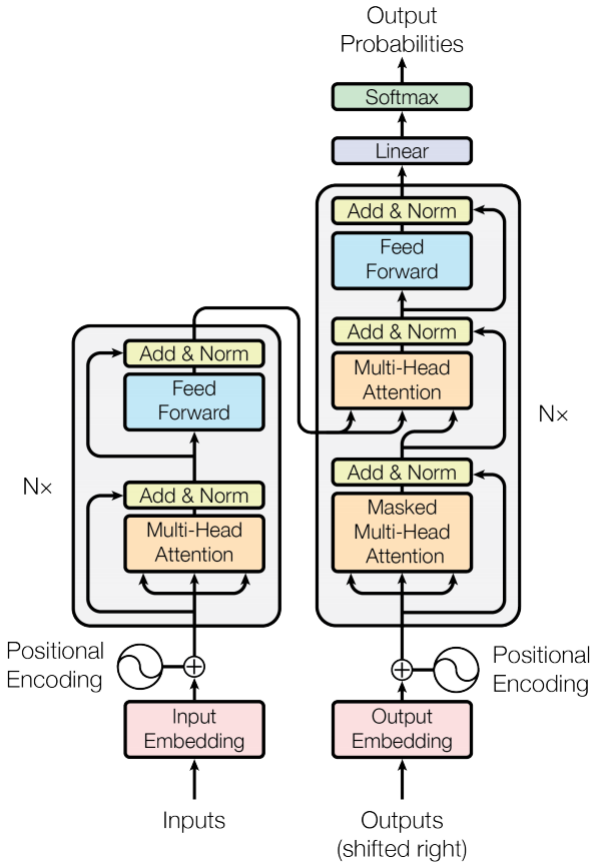
Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

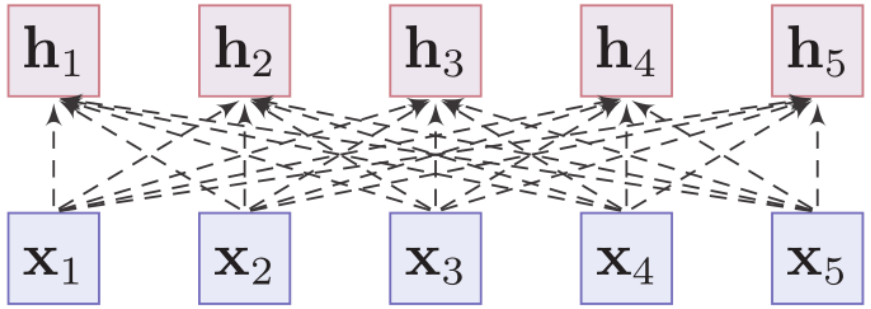
Illia Polosukhin* ‡ illia.polosukhin@gmail.com



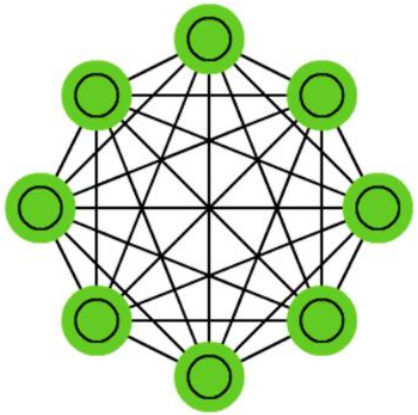
Vaswani, Ashish, et al. "Attention is All you Need." NIPS. 2017.

Transformer and GNN

- Transformer is a model built with self-attention module.



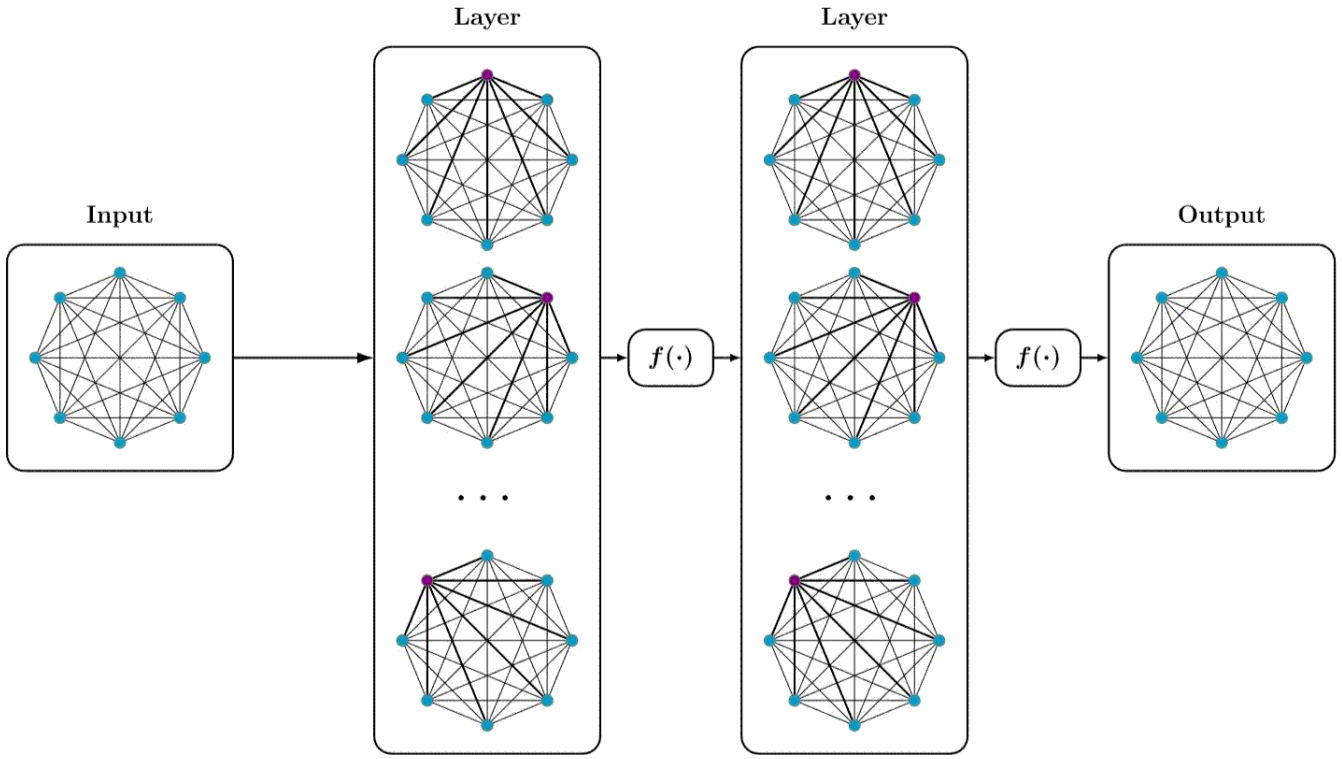
- Fully-connection



Weights α_{ij} are generated dynamically with attention mechanism

Graph Views of Transformers

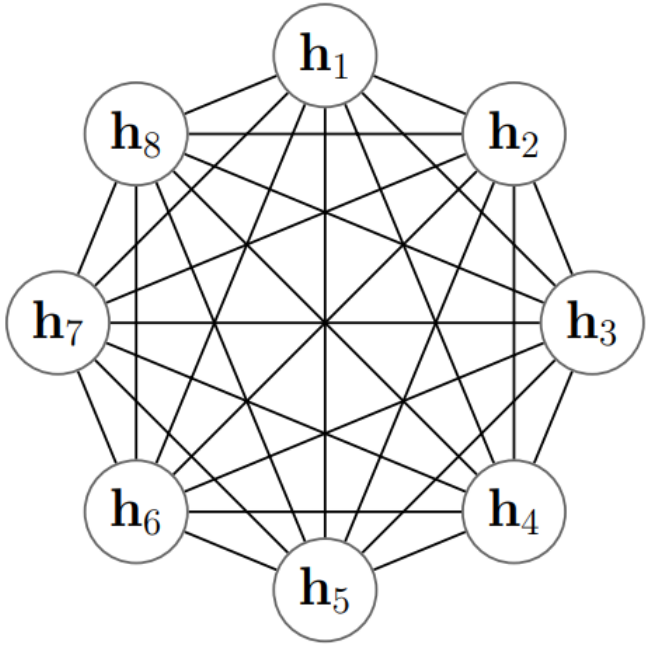
- Transformer is a model built with self-attention module.



Contents

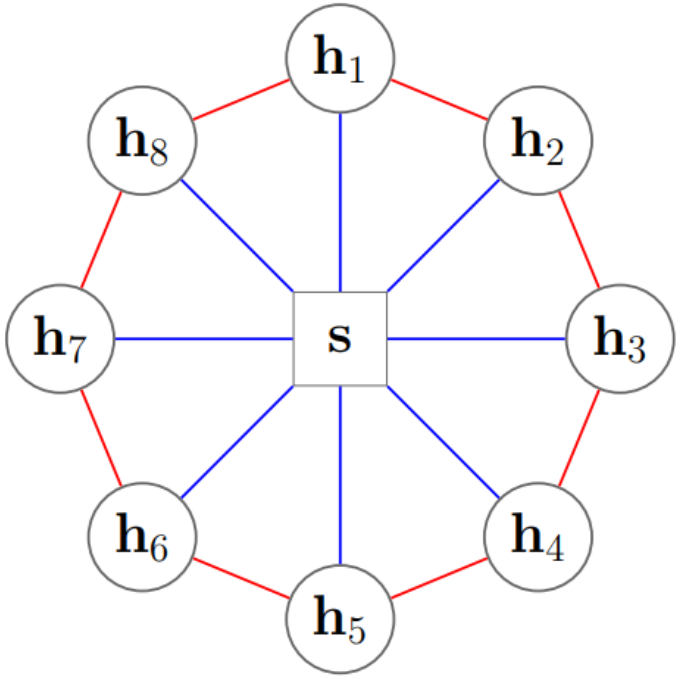
- Sparsification of self-attention
- Normalized Information Payload(NIP)
- Hypercube Transformer
- Experiments

Self-attention as Complete Graph



- Complexity : $\Theta(N^2)$
- Reduce complexity ----> Reduce the number of edges

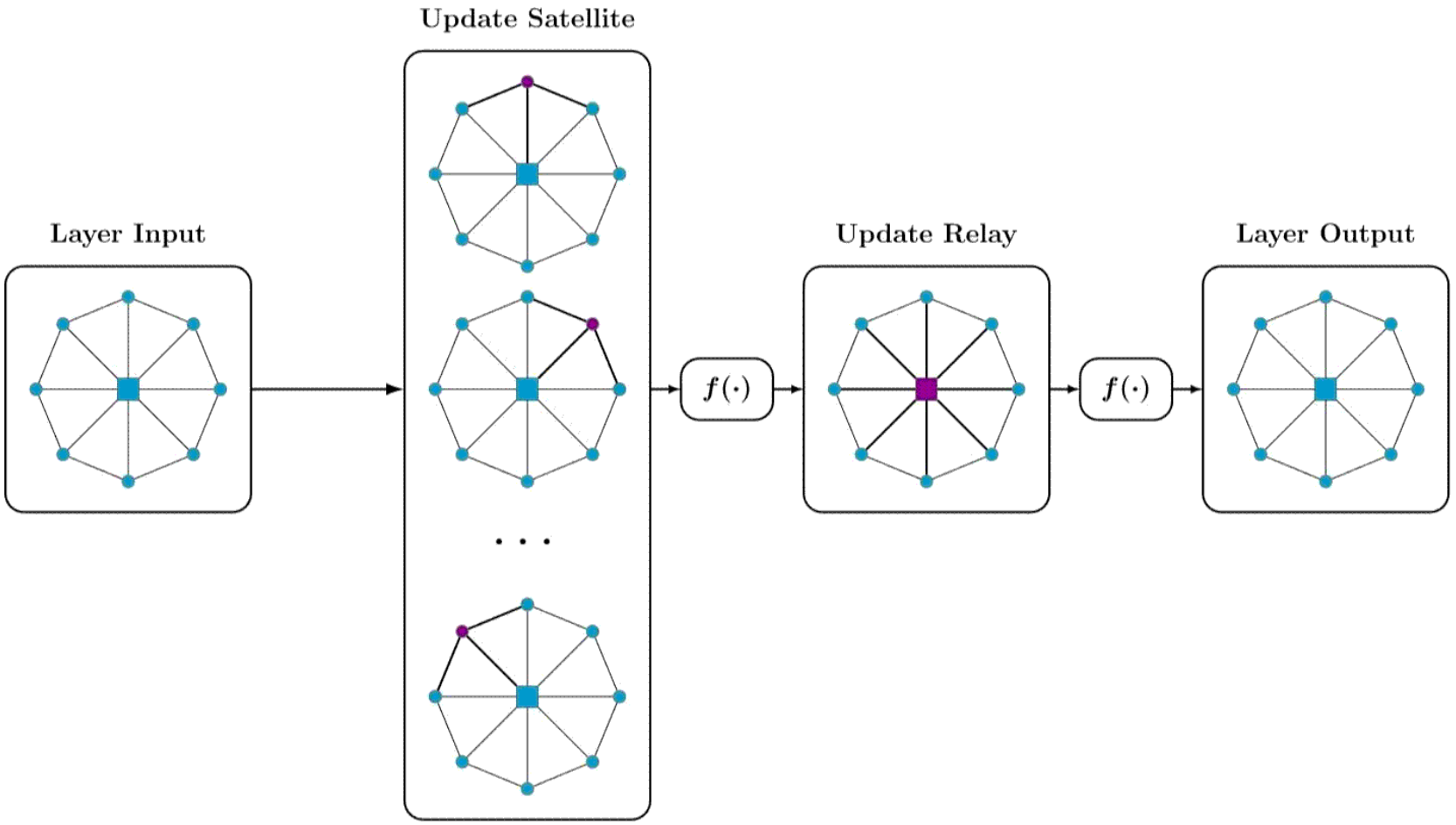
Star-Transformer



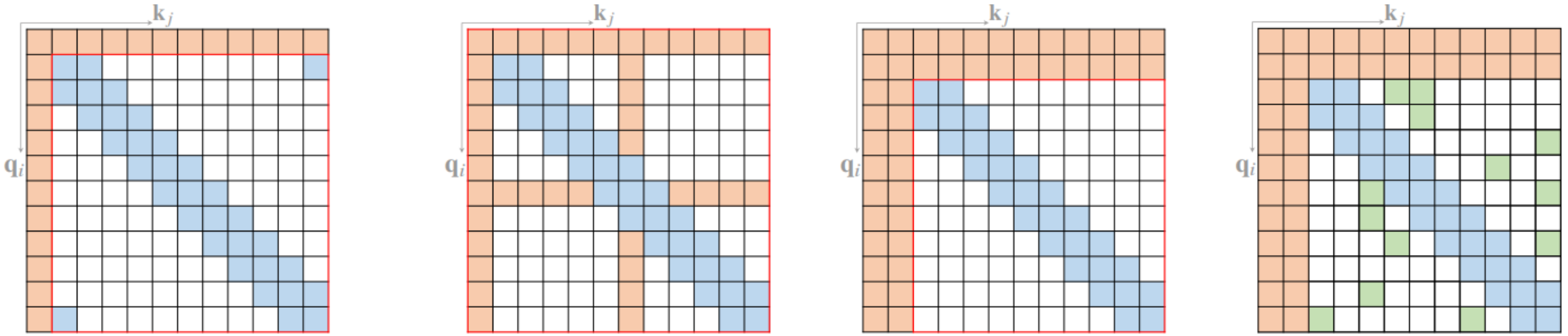
- Reduce Complexity to $\Theta(N)$.
- Preserve the capacity to capture both local composition and long-range dependency.

Qipeng Guo, Xipeng Qiu, Pengfei Liu, Yunfan Shao, Xiangyang Xue, Zheng Zhang. Star-Transformer, NAACL 2019, https://arxiv.org/pdf/1902.09113.pdf

Star-Transformer



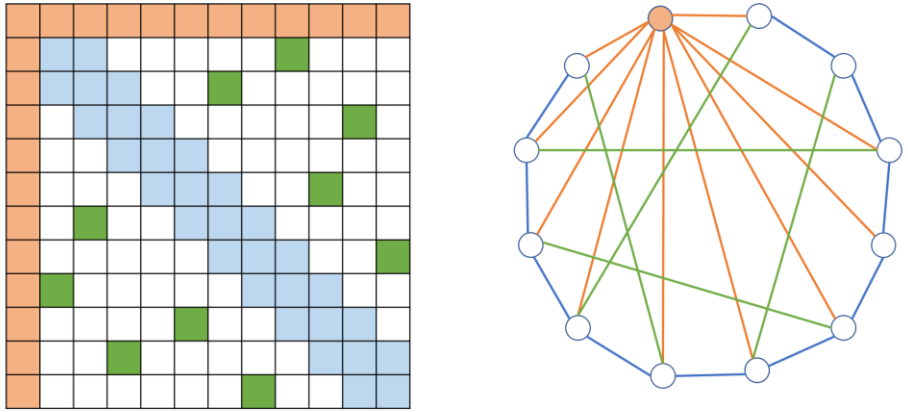
Empirical Sparse Transformers



(a) Star-Transformer (b) Longformer (c) ETC (d) BigBird

Lin, T., Wang, Y., Liu, X., and Qiu, X. A survey of transformers, 2021

Sparse Transformer: A Graph View



- Which property is important for those graphs serving as ground for self-attention?
- How dense do we need the graph to be in order to reduce complexity and at the same time remain performance?

Contents

- Sparsification of self-attention
- Normalized Information Payload(NIP)
- Hypercube Transformer
- Experiments

Two views of a graph

- Computational Complexity (CC)
  - Computational Complexity is the computation complexity required to allow the model to grab all interactions among tokens when using graph G for self-attention.
  - For complete graph, it's N.
- Information Payload (IP)
  - Information Payload. measuring how much information a graph can transfer when allowing the model to grab all interactions among tokens.
  - For complete graph, it's $\frac{1}{N-1}$.
- To better compare information transfer on different graphs, we define the Normalized Information Payload (NIP)

$$\mathrm{NIP}(G) := \frac{\mathrm{IP}(G)}{\mathrm{CC}(G)}.$$

Computational Complexity

- κ(G)
  - Given a G-attention layer, to make the whole model grab all interactions among tokens, we need to stack κ(G) G-attention layers.
  - Straightforwardly, κ(G) is the diameter of graph G.
  - For complete graph, it's 1.
- ρ(G)
  - For one G-attention layer, when the input sequence is fixed at length N, the Computational Complexity for one layer is proportional to the mean degree of G.
  - For complete graph, it's $\frac{N-1}{2}$.

$$\mathrm{CC}(G) := \rho(G) \times \kappa(G).$$

Information Payload

Definition 2.2. For one path $P_{ab} \in \mathcal{P}_{ab}$, the Information Payload of one path P_{ab}, denoted by $R(P_{ab})$, is defined as

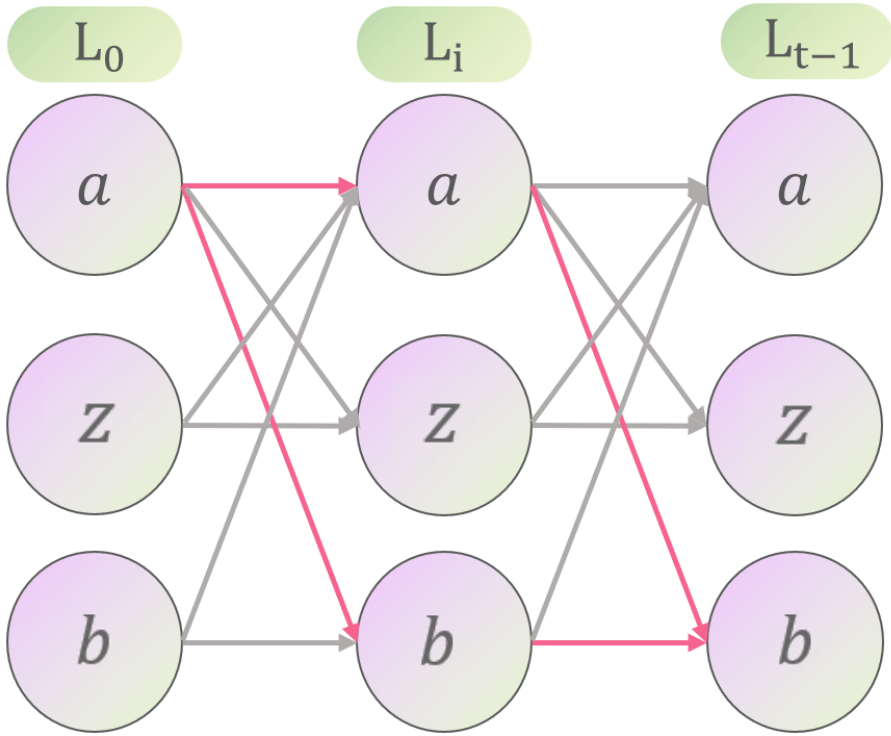
$$R(P_{ab}) := \prod_{v \in P_{ab} \ \& \ v \neq a} \frac{1}{deg(v)}.$$

Definition 2.3. The Information Payload between node pair (a, b), denoted by I_{ab} is sum of Information Payload of all paths that belong to $\mathcal{P}_{ab}$:

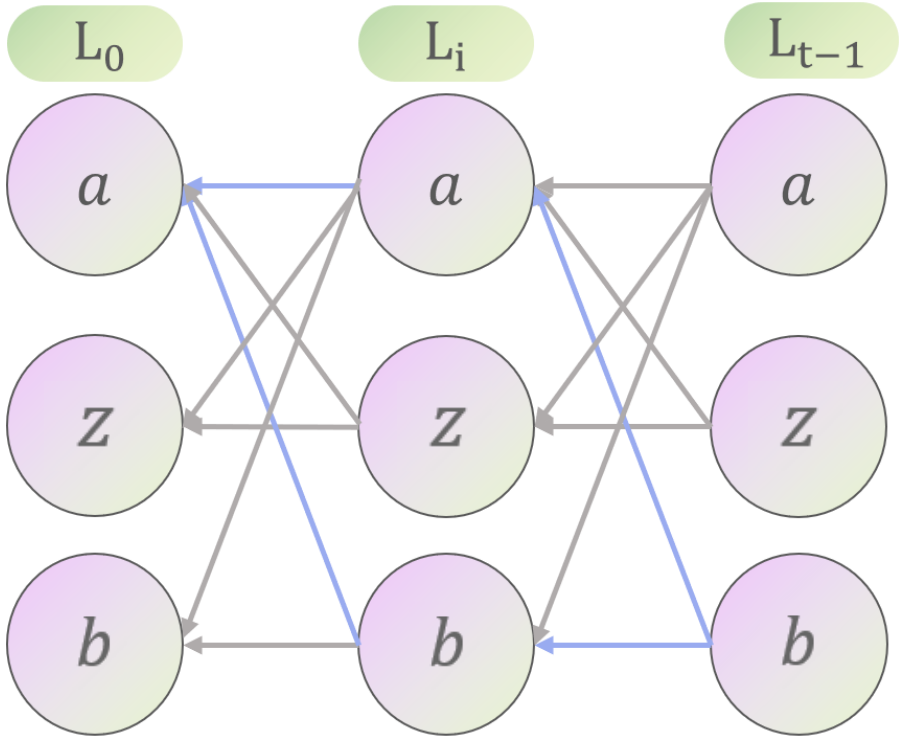
$$I_{ab} = \sum_{P_{ab} \in \mathcal{P}_{ab}} R(P_{ab}).$$

Closely related to Random Walk

Theorem 2.4. *Information Payload between two nodes I_{ab} equals to the probability of a random walk starts from node b that ends in node a at step $len(P_{ab})$.*



Attention forward



Random walk

Information Payload

Definition 2.5. The Information Payload for a graph G $\mathrm{IP}(G)$ is the smallest Information Payload I_{ab} between node pairs (a, b) whose distance is the diameter of the graph. Let Δ be the set of node pairs whose distance is the diameter of the graph, we have

$$\mathrm{IP}(G) := \min_{(a,b)\in\Delta} I_{ab}. \tag{5}$$

Normalized Information Payload (NIP)

Table 1. Normalized Information Payload for commonly used graphs, where w is the number of neighbors in ring lattice.

Type of graph	CC(G) ↓	IP(G) ↑	NIP(G) ↑
Complete	$\Theta(N)$	$\Theta\left(\frac{1}{N}\right)$	$\Theta\left(\frac{1}{N^2}\right)$
E-R random	$\Theta(\log^2 N)$	$\Theta\left(\frac{(N-2)!}{N^{\log N}(N-\log N)!}\right)$	$\Theta\left(\frac{(N-2)!/(N-\log N)!}{N^{\log N}\log^2(N)}\right)$
Tree	$\Theta(\log N)$	$\Theta(\frac{1}{N^{\log(9)}})$	$\Theta\left(\frac{1}{N^{\log(9)}\log N}\right)$
Star	$\Theta(1)$	$\Theta\left(\frac{1}{N}\right)$	$\Theta\left(\frac{1}{N}\right)$
Ring lattice + E-R random	$\Theta(\log N(\log N + w))$	$\Theta\left(\frac{(N-2)!}{(N+\frac{w}{\log N})^{\log N}(N-\log N)!}\right)$	$\Theta\left(\frac{(N-2)!/(N-\log N)!}{(N+\frac{w}{\log N})^{\log N}\log N(\log N+w)}\right)$
Ring lattice + Star (Longformer)	$\Theta(w)$	$\Theta\left(\frac{1}{Nw}\right)$	$\Theta\left(\frac{1}{Nw^2}\right)$
Ring lattice + Star + E-R random (BigBird)	$\Theta(\log N + w)$	$\Theta\left(\frac{1}{N(\log N+w)}\right)$	$\Theta\left(\frac{1}{N(\log N+w)^2}\right)$

Case Study

- Star Graph.
  - Extremely high NIP but limited performance in real-world tasks. Probably due to information interference through one global node.
- E-R random.
  - For the E-R random graph used in BigBird, if the probability of every edge to exist is p, the E-R random graph is highly possibly connected if $p > \frac{(1+\epsilon)lnN}{N}$. We use the lower bound and set $p = \frac{logN}{N}$.
- Ring Lattice.
  - Commonly used for grabbing local information but poor NIP.
- Longformer and Bigbird.
  - Random graph not necessarily make Bigbird have larger NIP than Longformer. Counterintuitive but reasonable, and fit Bigbird's experiments.

Contents

- Sparsification of self-attention
- Normalized Information Payload(NIP)
- Hypercube Transformer
- Experiments

How to design sparse graph with high NIP

- Complete graph : the shortest distance between two neighbors' neighbors (excluding two nodes themselves) equals to zero, meaning that every two neighbor has the same neighbor.

- Unknown graph: the shortest distance between two neighbors' neighbors (excluding two nodes themselves) equals to one. Much sparser while maintaining the connectivity of the graph.

- This Unknown graph is Hypercube.

Normalized Information Payload (NIP)

Table 1. Normalized I

Type of graph	NIP(G) ↑
Complete	$\Theta\left(\frac{1}{N^2}\right)$
E-R random	$\Theta\left(\frac{(N-2)!/(N-\log N)!}{N^{\log N}\log^2(N)}\right)$
Tree	$\Theta\left(\frac{1}{N^{\log(9)}\log N}\right)$
Star	$\Theta\left(\frac{1}{N}\right)$
Ring lattice + E-R random	$\left(\frac{(N-2)!/(N-\log N)!}{(N+\frac{w}{\log N})^{\log N}\log N(\log N+w)}\right)$
Ring lattice + Star (Longformer)	$\Theta\left(\frac{1}{Nw^2}\right)$
Ring lattice + Star + E-R random (BigBird)	$\Theta\left(\frac{1}{N(\log N+w)^2}\right)$
Hypercube	$\Theta\left(\frac{(\log N)!}{(\log N)^{\log N+2}}\right)$

er of neighbors in ring lattice.

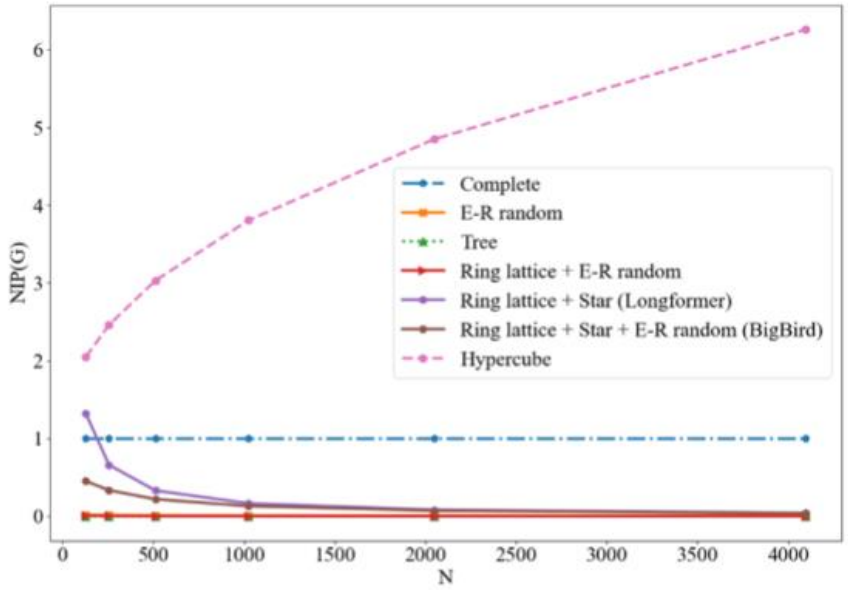


Figure 2. NIP(G) for graphs divided by complete graph in Table 1. We do not include star graph and ring lattice in this Figure because NIP(G) for star graph is too large. The w used for ring lattice is set to $\frac{N}{16}$ according to Longformer at length 4096.

Hypercube Transformer mapping

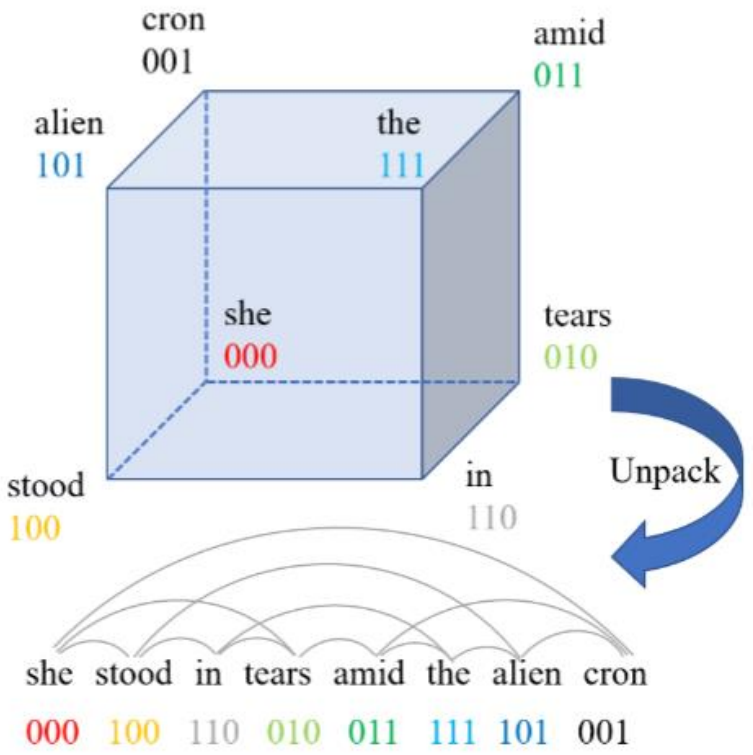


Figure 5. Unpacking a hypercube to a sequence. Tokens that are neighbors in hypercube are also neighbors in a sequence.

Algorithm 1 Binary representation of sequences

Input: sequence $S = (s_0, ..., s_{N-1})$
Output: Binary representation $X^N = X_0X_1X_2...X_i...X_{N-1}$
Initialize $X = (0, 1)$
repeat
  $Y = ()$
  for i **in** X **do**
    $Y.append(i << 1)$
  end for
  for i **in** reversed X **do**
    $Y.append(i << 1 + 1)$
  end for
  $X = Y$
until $len(X) >= N$
Output: $X^N = X[: N]$

If the dimension of binary numbers is k, the final binary representation for token with index i ($X_i := X_i^{k-1}X_i^{k-2}...X_i^1X_i^0$, $i \in [0, N-1]$) is given by

$$X_i^d = \left(\lfloor \frac{i \mod 2^{k-d+1}}{2^{k-d}} \rfloor + \lfloor \frac{i \mod 2^{k-d}}{2^{k-d-1}} \rfloor \right) \mod 2, \quad (6)$$

Hypercube Transformer mapping

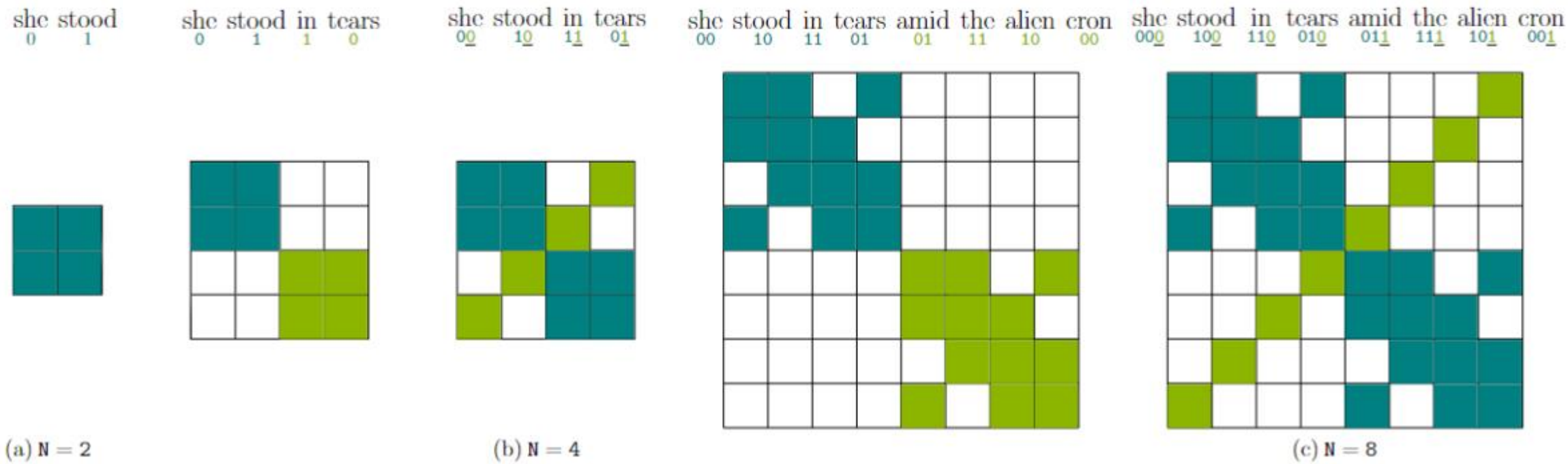


Figure 4. Iteratively mapping a sequence to a hypercube and its attention mask. Figure (a), (b) and (c) is the attention map for input sequences with length $N = 2, 4, 8$ respectively.

Contents

- Sparsification of self-attention
- Normalized Information Payload(NIP)
- Hypercube Transformer
- Experiments

Long-Range-Arena

Table 2. Performances for different graphs on Long-Range Arena

Graph #Length	Avg.	NIP(G)	SpeedUp
Complete	60.67	1×	1×
Star	59.95	-	-
Ring lattice + E-R random	59.69	$1.43e^{-10}$×	-
Ring lattice + Star (Longfc	60.00	$8.25e^{-2}$×	-
Ring lattice + Star + E-R r	59.99	$7.21e^{-2}$×	-
Hypercube	**62.07**	4.85×	15.8×

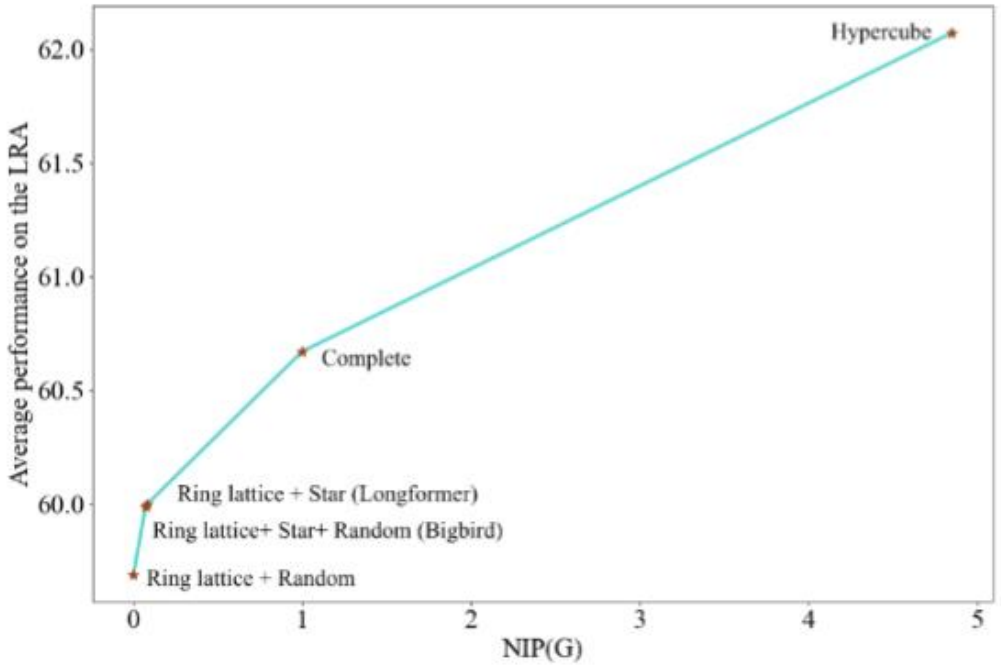


Figure 6. Average performance on the LRA benchmark can have strong proposition with our proposed Normalized Information Payload.

Block Sparsity

Theorem 3.1. *For block size* $b \leq \frac{N}{2}$*, larger block size makes star graph and hypercube have less Normalized Information Payload.*

Table 3. Performance of Hypercube Transformer with different block sizes.

Hypercube	Retrieval	Image
Block size 16	81.16	53.79
Block size 32	80.74	51.98
Block size 64	80.75	50.75

Pretraining and Finetuning on longer contexts

Table 5. Finetuning MLM on Wikitext103.

Model	Loss	Speedup
BERT_{128}	1.18	1×
CubeBERT_{128}	1.05	1.4×

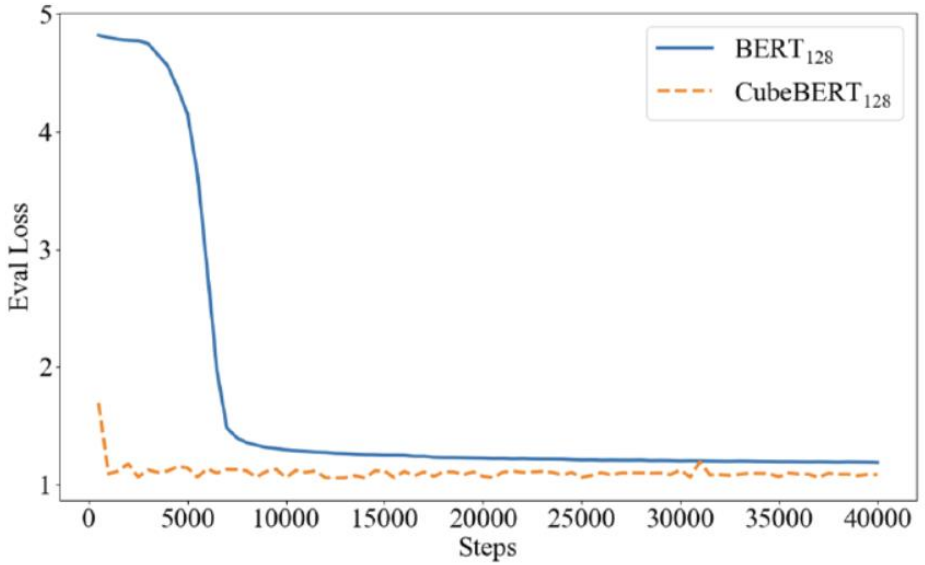


Figure 7. CubeBERT_{128} shows faster dropping rate of eval loss than BERT_{128} when finetuning on Wikitext103.

Pretraining and Finetuning on GLUE

Table 6. Performances on GLUE test sets. For our implementation, results for RTE, STS and MRPC are reported by first finetuning on the MNLI model instead of the baseline pretrained model.

	MNLI-m/mm	**QNLI**	**QQP**	**RTE**	**SST-2**	**MRPC**	**CoLA**	**STS-B**	**Avg.**	**Speedup**
#metric	Acc	Acc	F1	Acc	Acc	F1	Matthew's corr.	Spearman corr.		
#Examples	393k	105k	364k	2.5k	67k	3.7k	8.5k	7k		
BERT	86.0/85.2	92.6	72.0	78.3	94.5	89.9	60.9	87.5	83.0	1×
BERT_{128}	84.9/84.8	91.1	71.0	76.6	93.1	90.4	58.0	88.3	82.0	1×
CubeBERT_{128}	85.9/85.0	90.8	71.3	77.1	**95.3**	86.4	**61.5**	**87.6**	82.3	1.1×

Open Questions

- How to quantify the Information Interference of one node?

- Attention weights change with training, making the distribution not uniform. How to model the distribution of attention weights?
  - However, from analysis of BERT, some attention heads, especially in lower layers, have very broad attention, which means the uniform distribution assumption reasonable somehow.

Thank you for listening!