NAGphormer: A Tokenized Graph Transformer For Node Classification In Large Graphs

Jinsong Chen1,2,3,∗, Kaiyuan Gao2,3,∗, Gaichao Li1,2,3, Kun He2,3,†1
Institute of Artificial Intelligence
Huazhong University of Science and Technology， School of Computer Science and Technology
Hopcroft Center on Computing Science
chenjinsong,im_kai,gaichaolee,brooklet60}@hust.edu.cn

Code： https://github.com/JHL-HUST/NAGphormer.

—— ICLR 2023

2023.8.24 • ChongQing

Leibniz-Institut für Sozialwissenschaften

Reported by Jinyuan Zhang

1.Introduction

2.Overview

3.Methods

4.Experiments

gesis
Leibniz-Institut
für Sozialwissenschaften

Introduction

QUESTION:

1、Existing graph Transformers **hard to scale to large graphs**

2、**over-smoothing** and **over-squashing** problems, the negative influence of these inherent limitations cannot be eliminated completely.

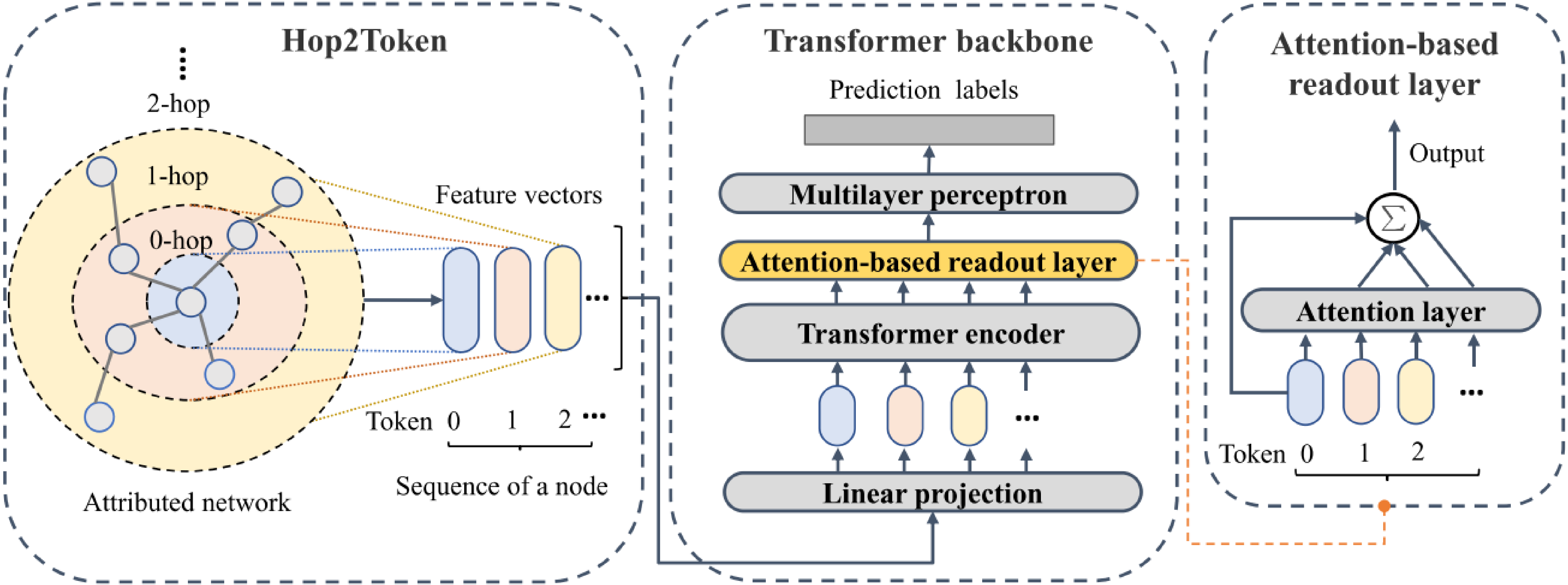
WORK:

1、**Hop2Token**， resulting in a sequence of token vectors that preserves neighborhood information for different hops， regard each node in the complex graph data as a sequence of tokens。

2、**NAGphorme**r, for the node classification task. In self-attention mechanism, it can learn more expressive node representations from the multi-hop neighborhoods

3、develop an **attention-based readout function** to adaptively learn the importance of different-hop neighborhoods to further boost the model performance.

Overview



Model framework of NAGphormer

Method

Graph Neural Network (GNN)

$$\mathbf{H}^{(l+1)} = \sigma(\hat{\mathbf{A}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}), \tag{1}$$

Decoupled Graph Convolutional Network (GCN)

$$\mathbf{Z} = \sum_{k=0}^{K} \beta_k \mathbf{H}^{(k)}, \mathbf{H}^{(k)} = \hat{\mathbf{A}}\mathbf{H}^{(k-1)}, \mathbf{H}^{(0)} = \boldsymbol{f}_\theta(\mathbf{X}), \tag{2}$$

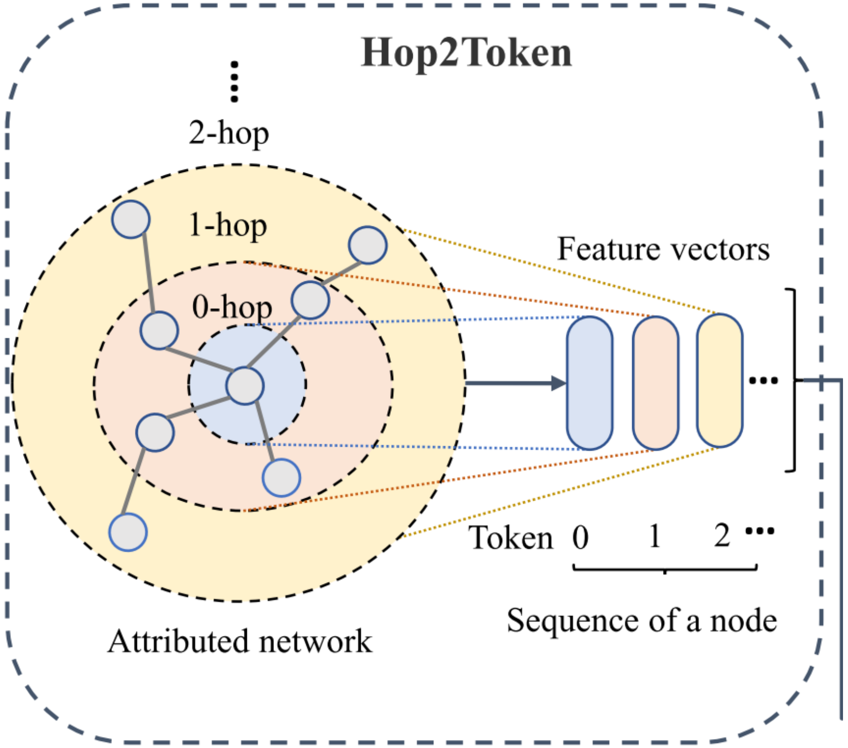
Hop2Token

$$\mathbf{X}' = \mathbf{X}\|\mathbf{U}. \tag{10}$$

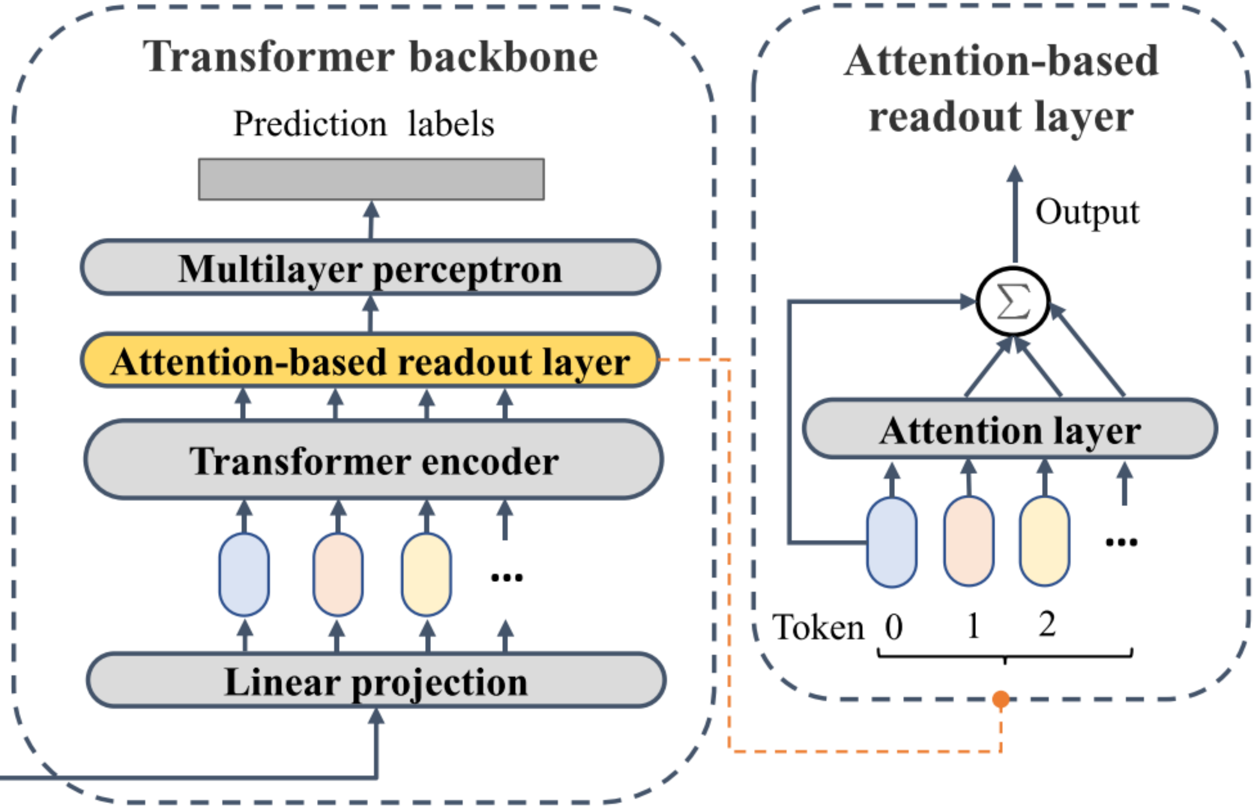
$$\mathbf{x}_v^k = \phi(\mathcal{N}^k(v)). \tag{5}$$

$$\mathbf{X}_k = \hat{\mathbf{A}}^k\mathbf{X}. \tag{6}$$

$$\mathcal{S}_v = (\mathbf{x}_v^0, \mathbf{x}_v^1, \ldots, \mathbf{x}_v^K)$$



Method



$$\mathbf{Z}_v^{(0)} = \left[\mathbf{x}_v^0\mathbf{E};\ \mathbf{x}_v^1\mathbf{E};\ \cdots;\ \mathbf{x}_v^K\mathbf{E}\right], \tag{7}$$

$$\mathbf{Z}_v'^{(\ell)} = \mathrm{MSA}\left(\mathrm{LN}\left(\mathbf{Z}_v^{(\ell-1)}\right)\right) + \mathbf{Z}_v^{(\ell-1)}, \tag{8}$$

$$\mathbf{Z}_v^{(\ell)} = \mathrm{FFN}\left(\mathrm{LN}\left(\mathbf{Z}_v'^{(\ell)}\right)\right) + \mathbf{Z}_v'^{(\ell)}, \tag{9}$$

$$\alpha_k = \frac{exp((\mathbf{Z}_0 \| \mathbf{Z}_k)\mathbf{W}_a^\top)}{\sum_{i=1}^{K} exp((\mathbf{Z}_0 \| \mathbf{Z}_i)\mathbf{W}_a^\top)}, \tag{11}$$

$$\mathbf{Z}_{out} = \mathbf{Z}_0 + \sum_{k=1}^{K} \alpha_k \mathbf{Z}_k. \tag{12}$$

Experiments

Method	Pubmed	CoraFull	Computer	Photo	CS	Physics
GCN	86.54 ± 0.12	61.76 ± 0.14	89.65 ± 0.52	92.70 ± 0.20	92.92 ± 0.12	96.18 ± 0.07
GAT	86.32 ± 0.16	64.47 ± 0.18	90.78 ± 0.13	93.87 ± 0.11	93.61 ± 0.14	96.17 ± 0.08
APPNP	88.43 ± 0.15	65.16 ± 0.28	90.18 ± 0.17	94.32 ± 0.14	94.49 ± 0.07	96.54 ± 0.07
GPRGNN	89.34 ± 0.25	67.12 ± 0.31	89.32 ± 0.29	94.49 ± 0.14	95.13 ± 0.09	96.85 ± 0.08
GraphSAINT	88.96 ± 0.16	67.85 ± 0.21	90.22 ± 0.15	91.72 ± 0.13	94.41 ± 0.09	96.43 ± 0.05
PPRGo	87.38 ± 0.11	63.54 ± 0.25	88.69 ± 0.21	93.61 ± 0.12	92.52 ± 0.15	95.51 ± 0.08
GRAND+	88.64 ± 0.09	71.37 ± 0.11	88.74 ± 0.11	94.75 ± 0.12	93.92 ± 0.08	96.47 ± 0.04
GT	88.79 ± 0.12	61.05 ± 0.38	91.18 ± 0.17	94.74 ± 0.13	94.64 ± 0.13	97.05 ± 0.05
Graphormer	OOM	OOM	OOM	92.74 ± 0.14	OOM	OOM
SAN	88.22 ± 0.15	59.01 ± 0.34	89.83 ± 0.16	94.86 ± 0.10	94.51 ± 0.15	OOM
GraphGPS	88.94 ± 0.16	55.76 ± 0.23	OOM	95.06 ± 0.13	93.93 ± 0.12	OOM
NAGphormer	$\mathbf{89.70 \pm 0.19}$	$\mathbf{71.51 \pm 0.13}$	$\mathbf{91.22 \pm 0.14}$	$\mathbf{95.49 \pm 0.11}$	$\mathbf{95.75 \pm 0.09}$	$\mathbf{97.34 \pm 0.03}$

Table 1: Comparison of all models in terms of mean accuracy $\pm$ stdev (%) on small-scale datasets. The best results appear in **bold**. OOM indicates the out-of-memory error.

Experiments

Table 2: Comparison of all models in terms of mean accuracy ± stdev (%) on large-scale datasets. The best results appear in **bold**.

Method	AMiner-CS	Reddit	Amazon2M
PPRGo	49.07 ± 0.19	90.38 ± 0.11	66.12 ± 0.59
GraphSAINT	51.86 ± 0.21	92.35 ± 0.08	75.21 ± 0.15
GRAND+	54.67 ± 0.25	92.81 ± 0.03	75.49 ± 0.11
NAGphormer	$\mathbf{56.21 \pm 0.42}$	$\mathbf{93.58 \pm 0.05}$	$\mathbf{77.43 \pm 0.24}$

Table 3: The accuracy (%) with or without structural encoding.

	Pubmed	CoraFull	CS	Computer	Photo	Physics	Aminer-CS	Reddit	Amazon2M
W/O-SE	89.06	70.42	95.52	90.44	95.02	97.10	55.64	93.47	76.98
With-SE	89.70	71.51	95.75	91.22	95.49	97.34	56.21	93.58	77.43
Gain	+0.64	+1.09	+0.23	+0.78	+0.47	+0.24	+0.57	+0.11	+0.45

Experiments

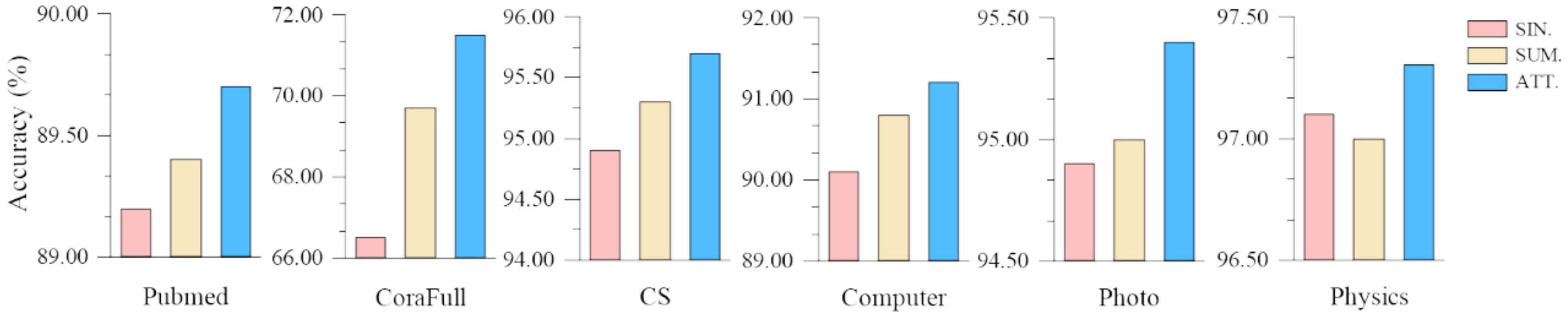


Figure 2: The performance of NAGphormer via different readout functions.

Experiments

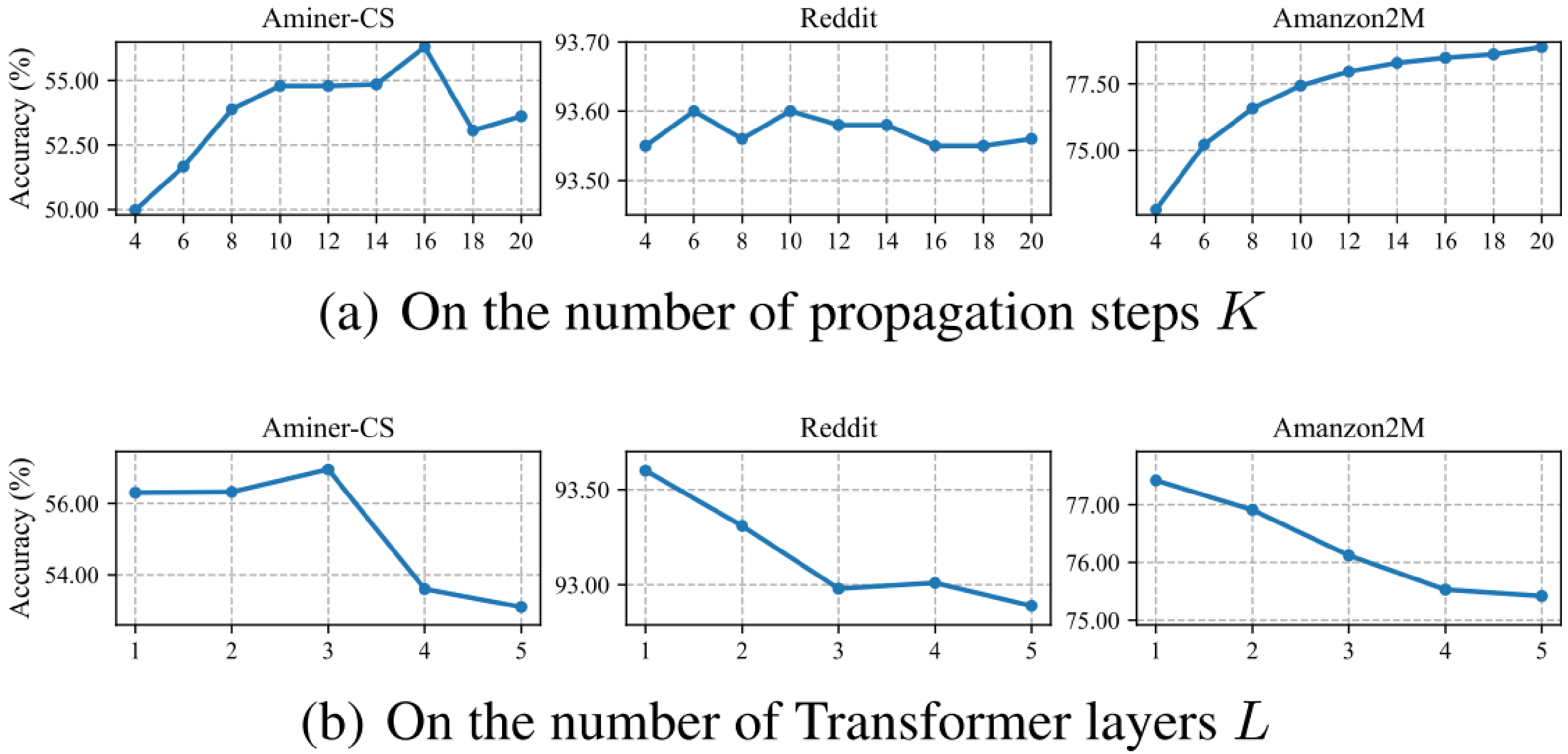


Figure 3: Performance of NAGphormer on different parameters.

THANKS