Sui YD, Wang X, Chen TL *et al.* Inductive Lottery Ticket Learning for Graph Neural Networks. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 33(1): 1–15 January 2018. DOI 10.1007/s11390-015-0000-0

Inductive Lottery Ticket Learning for Graph Neural Networks

Yong-Duo Sui[1] (隋勇铎), Xiang Wang[1,*] (王　翔), *Member, CCF,* Tianlong Chen[2] (陈天龙), Meng Wang[3] (汪萌), *Fellow, IEEE,* Xiang-Nan He[1,*] (何向南), *Member, CCF,* Tat-Seng Chua[4] (蔡达成)

[1] *School of Data Science, University of Science and Technology of China, Hefei 230027, China*

[2] *Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin 78712, U.S.A.*

[3] *School of Computer Science and Information Engineering, Hefei University of Technology, Hefei 230009, China*

[4] *School of Computing, National University of Singapore, Singapore*

E-mail: syd2019@mail.ustc.edu.cn; xiangwang@ustc.edu.cn; tianlong.chen@utexas.edu; wangmeng@hfut.edu.cn; hexn@ustc.edu.cn; dcscts@nus.edu.sg

Received June 28, 2022; accepted October 9, 2023.

Abstract Graph neural networks (GNNs) have gained increasing popularity, while usually suffering from unaffordable computations for real-world large-scale applications. Hence, pruning GNNs is of great need but largely unexplored. The recent work UGS studies lottery ticket learning for GNNs, aiming to find a subset of model parameters and graph structure that can best maintain the GNN performance. However, it is tailed for the transductive setting, failing to generalize to unseen graphs, which are common in inductive tasks like graph classification. In this work, we propose a simple and effective learning paradigm, Inductive Co-Pruning of GNNs (ICPG), to endow graph lottery tickets with inductive pruning capacity. To prune the input graphs, we design a predictive model to generate importance scores for each edge based on the input; to prune the model parameters, it views the weight's magnitude as their importance scores. Then we design an iterative co-pruning strategy to trim the graph edges and GNN weights based on their importance scores. Although it might be strikingly simple, ICPG surpasses the existing pruning method and can be universally applicable in both inductive and transductive learning settings. On ten graph-classification and two node-classification benchmarks, ICPG achieves the same performance level with 14.26%-43.12% sparsity for graphs and 48.80%-91.41% sparsity for the model.

Keywords lottery ticket hypothesis, graph neural networks, neural network pruning

1 Introduction

Graph neural networks (GNNs) [1–3] have become a prevalent solution for machine learning tasks on graph-structured data. Such success is usually ascribed to the powerful representation learning of GNN, which incorporates the graph structure into the representations, such as aggregating neural messages from the neighboring nodes to update the ego node's representation.

As the field grows, there is an increasing need of building deeper GNN architectures [4, 5] on larger-scale graphs [6]. While deepening GNNs shows potential on large-scale graphs, it also brings expensive computations due to the increased scale of graph data and model parameters, limiting their deployment in resource-constrained applications. Taking fraud detection in a transaction network as an example, the scale of user nodes easily reaches millions or even larger, making a GNN-detector model prohibitive to stack deep layers and predict malicious behaviors in real-time. Hence, pruning over-parameterized GNNs is of great need, which aims to answer the question: Can we co-sparsify the input graphs and model, while preserving

Regular Paper

This work was supported by the National Key Research and Development Program of China under Grant No. 2020YFB1406703, and the National Natural Science Foundation of China under Grant No. 9227010114.

*Corresponding Author

or even improving the performance?

Recently, a pruning approach, UGS [7], has been proposed to find graph lottery tickets (GLTs) — smaller subsets of model parameters and input graphs. At its core is Lottery Ticket Hypothesis (LTH) [8] speculating that any dense, randomly-initialized neural network contains a sparse subnetwork, which can be trained independently to achieve a matching performance as the dense network. Specifically, UGS employs trainable masks on each edge in the input graph and each weight in the model parameters, to specify their importance. When training the model with the masks, the strategy of iterative magnitude-based pruning (IMP) [8] is used to discard the edges and weights with the lowest mask values at each iteration.

Despite the effectiveness, there exist the following limitations: (1) UGS focuses solely on providing transductive graph masks by generating a painstakingly customized mask for a single edge individually and independently. That is, the edge masks are limited to the given graph, making UGS infeasible to be applied in the inductive setting since the edge masks hardly generalize to unseen edges or entirely new graphs. (2) Applying a mask for each edge alone only provides a local understanding of the edge, rather than the global view of the entire graph (e.g., in node classification) or multiple graphs (e.g., in graph classification). Moreover, the way of creating trainable edge masks will double the parameters of GNNs, which violates the purpose of pruning somehow. As a result, these edge masks could be suboptimal to guide the pruning. (3) The unsatisfactory graph pruning will negatively influence the pruning of model weights. Worse still, low-quality weight pruning will amplify the misleading signal of edge masks in turn. They influence each other and form a vicious circle. We ascribe all these limitations of UGS to its transductive nature. Hence, conducting combinatorial pruning in the inductive setting is crucial to high-quality winning tickets.

In this work, we emphasize the inductive nature within the combinatorial pruning of input graphs and GNN parameters and present our framework, Inductive Co-Pruning of GNNs (ICPG). It is an extremely simple but effective pruning framework that is applicable to any GNN in both inductive and transductive settings. Specifically, for the input graphs, we design a predictive model, AutoMasker, which learns to generate edge masks from the observed graphs. It is parameterized with an additional GNN-based encoder, whose parameters are shared across the population of observed graphs. As a consequence, AutoMasker is naturally capable to specify the significance of each edge and extract core subgraphs from a global view of the entire observations. For the model parameters, we simply exploit the magnitude of a model weight to assess whether it should be pruned, rather than training an additional mask. Having established the edge masks and weight magnitudes, we can obtain high-quality GLTs by pruning the lowest-mask edges and lowest-magnitude weights. Experiments on ten graph classification and two node classification datasets consistently validate our framework ICPG by identifying high-quality GLTs. Moreover, we inspect the GNN-level and graph-level transferability, which promises for deploying ICPG in the pre-training and fine-tuning paradigm to save the computational cost. The visualizations show that ICPG always retains decisive subgraphs, such as edges located on digital pixels in MNIST graphs, which further illustrates rationality and explainability.

In all, our main contributions can be summarized as follows.

We introduce ICPG, an innovative pruning framework, capable of pruning both the GNN model and input graphs, which excels at identifying high-quality GLTs across diverse graph representation tasks in both inductive and transductive settings.

We have validated ICPG's capacity to find GLTs in datasets of various scales through extensive experiments, while maintaining performance with a range of graph sparsity from 22.62% to 43.12% and GNN sparsity from 67.23% to 91.41%.

We demonstrate that ICPG offers transferability at both the GNN and graph levels, resulting in im-

proved performance and lower computational costs in downstream tasks. This is substantiated by thorough comparisons, analyses, and visual inspections that validate its effectiveness, applicability, and explainability.

2 Related Work

Graph Neural Networks (GNNs) [1–3,9,10] have emerged as a powerful tool for learning the representation of graph-structured data. The great success mainly comes from the structure-aware learning, which follows the iterative message-passing scheme. Specifically, we denote an undirected graph by $G = (A; X)$ with the node set V and edge set E. $A \in \{0;1\}^{|\mathcal{V}|\times|\mathcal{V}|}$ is the adjacency matrix, where $A[i;j] = 1$ denotes the edge between node v_i and node v_j, otherwise $A[i;j] = 0$. $X \in \mathbb{R}^{|\mathcal{V}|\times d}$ is the matrix of node features, where $x_i = X[i;:]$ is the d-dimensional feature of the node $v_i \in V$. Given a K-layer GNN, its k-th layer generates the representation of node v_i as in (1) and (2).

$$a_i^{(k)} = AGGREGATION^{(k)}(\{h_j^{(k-1)} | j \in \mathcal{N}(i)\}); \quad (1)$$

$$h_i^{(k)} = COMBINE^{(k)}(h_i^{(k-1)}; a_i^{(k)}); \quad (2)$$

where $h_i^{(k)}$ and $a_i^{(k)}$ are the representation of node v_i and the message aggregated from its neighbor nodes set $\mathcal{N}(i)$, respectively; the *AGGREGATION* and *COMBINE* operators are the message passing and update functions, respectively. After propagating through K layers, we get the final representations of nodes, which facilitate downstream node-level tasks, such as node classification and link prediction. As for graph-level tasks like graph classification and graph matching, we further hire the *READOUT* function to generate the representation of the whole graph G, as in (3).

$$Z_G = READOUT(\{h_i^{(k)} | v_i \in V; k \in \{1;:::;K\}\}): \quad (3)$$

Various GNNs [2,9,10] adopt different *AGGREGATION*, *COMBINE* and *READOUT* functions to refine the desired information.

Lottery Ticket Hypothesis (LTH) [8] states that a sparse subnetwork exists in a dense randomly-initialized network that can be trained to achieve comparable performance to the full models. LTH is explored in many fields such as computer vision and natural language processing [11–15]. Recently, UGS [7] extends the LTH to the GNNs, proposing the Graph Lottery Ticket (GLT), which includes subgraph and subnetwork pairs that can be trained independently to reach comparable performance to the dense pairs. However, due to the transductive nature of graph-specific masks, UGS [7] cannot develop in inductive learning settings. To address this issue, we have incorporated AutoMasker into our approach. This tool possesses the capability to learn the importance of each edge from training graphs on a global scale and predict significance scores for newly introduced graphs. By being both graph-agnostic and inductive, AutoMasker effectively surmounts the limitations traditionally associated with graph-specific masks, thus paving the way for novel advancements within inductive learning settings.

¦ 9@e0 ou Comprehensive Comparisons in the Inference Stage

Method	Sparse Graph		Sparse Model
	Transductive	Inductive	
SGAT [Eo£F]	3	7	7
NeuralSparse [Eo¢F]	3	3	7
GraphSAGE [E©oF]	3	3	7
DropEdge [EoPF]	3	3	7
SGCN [EokF]	3	7	7
GEBT [E©©F]	3	7	3
UGS [E¢F]	3	7	3
ICPG (Ours)	3	3	3

Graph Sparsification and Sampling aim to find core subgraphs in graph learning. Numerous strategies [16–27] are proposed to achieve efficient training or inference. SGAT [16] adopts sparse attention to remove edges. NeuralSparse [17] utilizes a DNN to identify task-irrelevant edges. Sampling-based methods [18–21] sample and aggregate features from a node's local neighborhood. DropEdge [18] randomly drops edges from the input graph, which can be seen as a data augmenter. Another research line selects subgraphs in an optimization way. SGCN [19] and GEBT [22] adopt the ADMM optimization algorithm to sparsify the adjacency matrix. UGS [7] utilizes trainable masks to prune graphs. Unfortunately, these methods either fail to utilize sparse

graphs in the inductive inference stage or do not use sparse GNNs for efficient inference. Distinct from them, ICPG endows GNNs with inductive sparsification capacity, which can universally work in both transductive and inductive settings with both sparse graphs and models. We make comprehensive comparisons with the above methods in Table 1.

3 Preliminaries

In this section, we first briefly introduce the inductive graph learning. Then we formulate the task of learning graph lottery tickets under inductive setting.

3.1 Inductive Graph Learning

Before entering our method, we first clarify the inductive learning settings of our work. Compared with inductive graph learning, transductive graph learning denotes that unlabeled test data can be used in the training process. For example, in semi-supervised node classification tasks [1], training and test nodes form an entire graph. During model training, we need to take the full graph data as input and predict the class of test nodes based on all node features (including test node features), all edges, and labels of training nodes. Hence, all information (except labels) on the test data is available during training. In contrast, inductive graph learning means that all information of the test graph is not available during the training process. For example, in graph classification tasks, we use the training data at hand to train GNN models, hoping that the models can effectively generalize to the unseen test data. Compared with the transductive graph learning setting, inductive graph learning cannot utilize any information from test data. Therefore, inductive learning requires better generalization ability of the model. Unfortunately, UGS [7] designs learnable weights for all edges of an entire graph. This training strategy requires that the topological structure of the graph data is fixed and invariant between the training and inference stage. Hence, UGS [7] is only possible to apply to the setting of transductive graph learning, while it cannot apply to inductive graph learning.

3.2 Inductive Graph Lottery Ticket (GLT)

Without loss of generality, we consider the task of graph classification as an example. Given a GNN classifier $f(\ ;\ _{g_0})$, it starts from the randomly-initialized parameters $_{g_0}$ before training and arrives at the well-optimized parameters $_{g}$ after training. Once trained, it takes any graph $G = (A; X)$ as the input and yields a probability distribution over C classes $\hat{y} = f(G;\ _{g})$.

Learning GLTs aims to make the input graph G and the model weights $_{g_0}$ sparse to reduce the computational costs, while preserving the performance. Formally, it aims to generate two masks $m_{\mathcal{G}}$ and m_{Θ}, which are applied on G and $_{g_0}$ correspondingly, so as to establish the sparser input graph $G' = (m_{\mathcal{G}}\ \ A; X)$ and initialized weights $'_{g_0} = m_{\Theta}\ \ _{g_0}$. Hereafter, through retraining the subnetwork $f(\ ;\ '_{g_0})$ on the sparse versions $\{G'\}$ of training graphs, we can get the new converged parameters $'_{g}$. If the well-optimized subnetwork can achieve comparable performance with full graphs and network, we term the pair of G' and $f(\ ;\ '_{g_0})$ as a GLT. Although a recent study, UGS [1], proposes to learn the GLTs, it focuses solely on the transductive setting but leaves the inductive setting untouched. Specifically, it assigns a trainable mask to each edge of the input graph and trains such graph-specific masks individually and independently. As a consequence, these edge-dependent masks are limited to the given graph, hardly generalizing to unseen edges or entirely new graphs. Distinct from UGS, we aim to uncover GLTs in the inductive learning setting.

4 Methodology

In this section, we propose a novel pruning framework, named Inductive Co-Pruning of GNNs (ICPG), to find the GLTs. We first introduce the key component in ICPG, named AutoMasker. Then we present our inductive strategy of co-pruning the input graphs and model parameters.

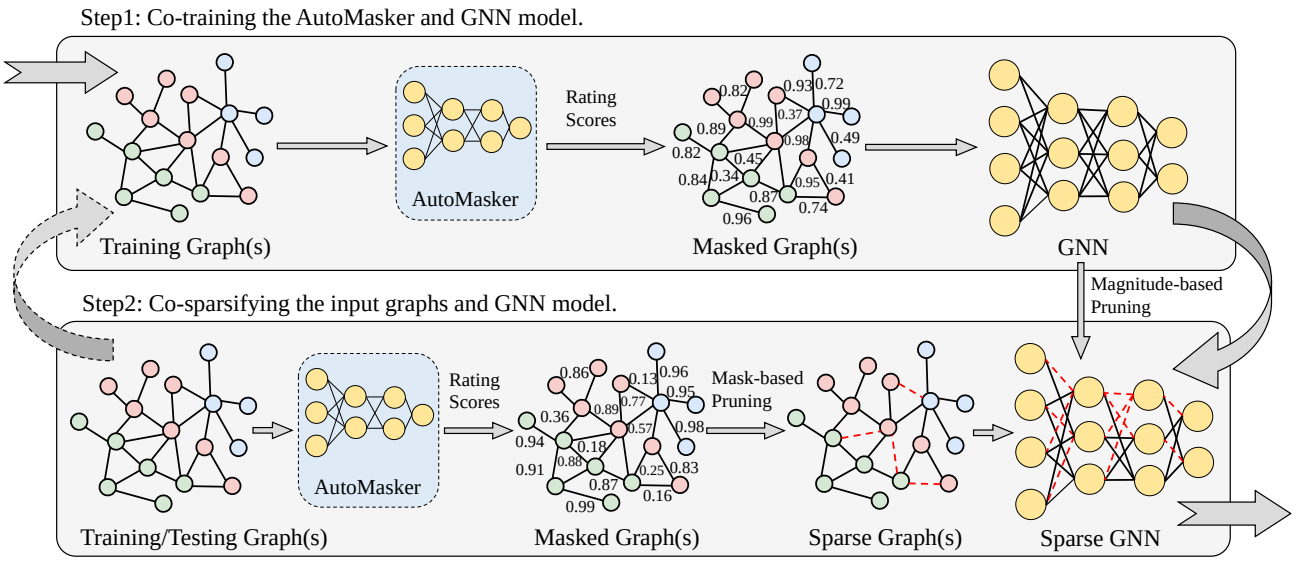


Fig.1. ICPG framework to find the GLTs.

4.1 AutoMasker

Instead of assigning a mask to a single edge, our idea is extremely simple: we take a collection of graph instances and design a trainable model to learn to mask edges collectively. The key ingredient of this model is an additional GNN-based model, termed AutoMasker, whose parameters are shared across the population of observed graphs. Here we represent AutoMasker as the combination of a graph encoder and a subsequent scoring function. Formally, given a graph $G = (A; X)$, AutoMasker applies a GNN-based graph encoder $g(\ )$ to create representations of all nodes as:

$$H = g(A; X);$$

where H 2 $\mathsf{R}^{|\mathcal{V}\times d|}$ stores d-dimension representations of all nodes, whose i-th row h_i denotes the representation of node v_i; $g(\ )$ is a GNN following the message-passing paradigm in (1). To assess the importance score of edge $(i; j)$ between node v_i and node v_j, AutoMasker builds a multi-layer perceptron (MLP) upon the concatenation of node representations h_i and h_j, which yields the score $_{ij}$. In what follows, the sigmoid function $(\ )$ projects $_{ij}$ into the range of $(0; 1)$, which represents the probability of edge $(i; j)$ being the winning ticket. The scoring function is represented as follows:

$$s_{ij} = (\ _{ij}); \qquad _{ij} = \text{MLP}([h_i; h_j]):$$

By employing the scoring function over all possible edges, we are able to collect the matrix of edge masks $s_{\mathcal{G}}$, where $s_{\mathcal{G}}[i; j] = s_{ij}$ if edge $(i; j)$ holds, otherwise $s_{\mathcal{G}}[i; j] = 0$. In a nutshell, we summarize the AutoMasker function as:

$$s_{\mathcal{G}} = \text{AutoMasker}(G; \ _a); \tag{4}$$

where $_a$ is the parameter of AutoMasker, covering the parameters of the GNN encoder and the MLP.

Although the key ingredient of AutoMasker is simple, it has several conceptual advantages over UGS.

Algorithm 1 Mask & Magnitude-based Pruning

Input: D, $f(\ ; \ _{g_0})$, $AutoMasker(\ ; \ _{a_0})$, M, m_Θ, Epoch T.
Output: Sparsified masks $f m'_{\mathcal{G}_i} g_{i=1}^N$, m'_Θ.

```
for t = 0 to T  1 do
  for G_i 2 D and m_{G_i} 2 M do
    G_i  (m_{G_i}  A_i; X_i)
    s_{G_i}  AutoMasker(G_i;  _{a_t})
    G_i  (s_{G_i}  A_i; X_i)
    Forward f(G_i; m_Θ  _{g_t})
    Backward to update  _{a_{t+1}},  _{g_{t+1}}
  end for
end for
for G_i 2 D do
  s_{G_i}  AutoMasker(G_i;  _{a_T})
  Set p_g = 5% of the lowest mask values in s_{G_i} to
  0 and others to 1, creating m'_{G_i}.
  Prune p_θ = 20% of the lowest magnitude parameters in  _{g_T}, creating m'_Θ.
end for
```

1) Global view. Although edge masks derived from UGS might preserve the fidelity to local importance, they do not help to delineate the general picture of the whole graph population. Distinct from UGS, AutoMasker takes a global view of making decisions. Specifically, for instance-level, AutoMasker adopts GNN as its backbone. Because of the message-passing mechanism of GNN, AutoMasker can fully consider the topology information of the entire graph data. For the dataset-level, all graph data share an AutoMasker, therefore it can make decisions for each edge in each graph data from a global perspective by observing all graph data in the dataset. As edges usually collaborate to make predictions, rather than working individually, they form a coalition like the functional groups of a molecule graph, the community of a social network. Hence, AutoMasker will learn the invariant and stable patterns in training data and can well transfer the learned patterns to the unseen test data.

2) Lightweight edge masks. When using UGS to prune graph data with millions of edges or nodes, the cost of assigning local edge masks one by one will be prohibitive with such a large-scale dataset in real-world scenarios. Moreover, UGS introduces additional parameters, whose scale remains the same as the edge number $\sum_{\mathcal{G}} jEj$ and is much larger than the original parameters being pruned. Hence, it somehow violates the purpose of pruning. In our AutoMasker, the additional parameter is $_a$ in (4), which remains invariant across the change of data scale.

3) Generalization. In contrast to UGS, AutoMasker can generalize the mechanism of mask generation to new graphs without retraining, making it more efficient to prune unseen and large-scale graphs. Hence, it makes ICPG more scalable and flexible for pruning in diverse real-world graph learning tasks or applications. In addition, we also conduct extensive experiments to verify this point.

4.2 Inductive Co-Pruning Strategy

Here we present Inductive Co-Pruning of GNNs (ICPG) to learn the GLTs. Fig.1 demonstrates its overview, which consists of the following two steps:

Step 1: Co-Training AutoMasker and the GNN Model of Interest. Given an input graph $G = (A; X)$, AutoMasker first generates the edge mask $s_{\mathcal{G}}$ via (4). Then we apply $s_{\mathcal{G}}$ to the adjacency matrix A to create the soft-masked graph $G_s = (s_{\mathcal{G}} \quad A; X)$, which fully reflects AutoMasker's decision for the importance of each edge, such that less important edges are prone to have lower mask values. Finally, we feed the soft-masked graph into the GNN model to co-train the AutoMasker and the model. The GNN model adopts the masked graph to learn the representation and make predictions, which can be viewed as the supervision signals to guide the AutoMasker to achieve a more accurate decision. The detailed co-training process is shown in Algorithm 1. When the training is done, we conduct Step 2 to perform the pruning.

Algorithm 2 Finding GLTs by ICPG

Input: Graphs $D = fG_i = (A_i; X_i)g_{i=1}^N$, $f(; \ _{g_0})$, *AutoMasker*$(; \ _{a_0})$, sparsity levels s_d, s_θ.
Output: GLT $fG_i' = (m_{\mathcal{G}_i} \quad A_i; X_i)g_{i=1}^N$, $f(; m_\Theta \ _{g_0})$.

```
Initialize masks set M  fm_{G_i}  A_i g_{i=1}^N
Initialize GNN mask m_Θ  1 2 R^{‖Θ_{g_0}‖_0}
while the sparsity of M < s_d, m_Θ < s_θ do
    Sparsify GNN f(; _{g_0}) with m_Θ and graphs fG_i = (A_i; X_i)g_{i=1}^N with the mask set M and get the new masks as presented in Algorithm 1.
    Update M  fm_{G_i}  m'_{G_i} g_{i=1}^N
    Update m_Θ  m'_Θ
    Rewind AutoMasker's weight to _{a_0}.
    Rewind GNN's weight to _{g_0}.
end while
```

Step 2: Co-Sparsifying the Input Graphs and GNN Model. Having obtained the well-trained AutoMasker and GNN, we can apply the learned knowledge to co-sparsify the graphs and model. For graphs, AutoMasker predicts the importance score (e.g., mask value) for each edge. Then the edges of a certain graph are sorted based on their mask values, and the edges with p_g ratio of the lowest-masks are pruned to obtain the mask $m_{\mathcal{G}}$. For GNN, we sort the parameters based on the their magnitude and prune p_θ ratio of the lowest-magnitude

parameters to obtain the binary model mask m_Θ. Under the current sparsity, we now successfully obtain the sparsified graph $G' = (m_G \quad A; X)$ and the sparsified GNN mask m_Θ. Finally, we need to check whether the sparsity meets our condition. If the sparsity is satisfied, the algorithm is completed; if not, we reuse the found GLT to update the original graphs and GNN model, and iteratively run Step 1 and Step 2 (dotted arrow in Fig.1) until the condition is met.

In summary, Algorithm 2 offers the detailed process of ICPG, where the sparsity levels s_θ and s_d refer to the proportions of model weights and graph edges that need to be pruned. Following LTH [8] and UGS [7], we also adopt an iterative pruning strategy to locate GLTs. In Algorithm 2, it will conduct Algorithm 1 to prune a certain proportion of graph edges and model weights. In our experiments, for graph data, we prune 5% of the edges each time, and for the model, we prune 20% of the weights each time. So we need to execute Algorithm 1 several times to achieve the given sparsity levels.

5 Experiments

In this section, we conduct extensive experiments to validate the effectiveness of ICPG. We first introduce the experimental settings and explore the existence of GLTs in graph classification and node classification. Then, we demonstrate the practicability, such as transferability, performance, and computational cost saving. Finally, more ablation studies and visualizations are provided.

5.1 Experimental Settings

Datasets. For graph classification, we adopt TU datasets [28–30], including biological graphs (NCI1, MUTAG), social graphs (COLLAB, RED-B, RED-M5K, RED-M12K). We also use superpixel graphs (MNIST, CIFAR-10) [31,32], and Open Graph Benchmark (ogbg-ppa and ogbg-code2) [6]. We use these graph classification datasets for inductive graph learning. For node classification, we choose a transductive learning dataset, Cora, and an inductive learning dataset, PPI. The detailed statistics of the datasets are shown in Table 2, where "#" refers to the number and "Avg." means the average number.

Models. We adopt the same model architecture for the GNN backbone and GNN encoder in AutoMasker. For graph classification tasks and Cora, we adopt the GCN [1] model. For PPI, we choose GAT [2] to achieve a better baseline performance.

¦9@e0 ©u Datasets Statistics

Dataset	#Graph	Avg. Nodes	Avg. Edges	Avg. Degree	#Class
NCI1	4,110	29.87	32.30	1.08	2
MUTAG	188	17.93	19.79	1.10	2
COLLAB	5,000	74.49	2457.78	32.99	3
RED-B	2,000	429.63	494.07	1.15	2
RED-M5K	4,999	508.52	594.87	1.17	5
RED-M12K	11,929	391.41	456.89	1.16	11
MNIST	70,000	70.57	564.56	8.00	10
CIFAR-10	60,000	117.63	941.04	8.00	10
ogbg-ppa	158,100	243.40	2,266.10	9.31	37
ogbg-code2	452,741	125.20	124.20	0.99	-
Cora	1	2708.00	5429.00	2.00	7
PPI	24	2372.67	34113.16	14.38	121

Training Settings. Here we provide the detailed training settings of the proposed ICPG. All training hyper-parameters such as epoch, learning rate (LR), optimizer, batch size, and weight decay are summarized in Table 3. For the devices, we adopt the NVIDIA GeForce RTX 3090 (24GB GPU) to conduct all our experiments. To help readers easily reproduce our results, we also provide the code of our work¬.

¦9@e0 ¯u Training Details of ICPG

Dataset	Epoch	LR	Optimizer	Batch Size	Weight Decay
TU	100	0.001	Adam	128	0
Superpixel	100	0.001	Adam	128	0
ppa	100	0.001	Adam	32	0
code2	25	0.001	Adam	128	0
Cora	200	0.01	Adam	1	0.0005
PPI	100	0.005	Adam	1	0

5.2 GLTs in Graph Classification Tasks

We first conduct experiments to find the GLTs in graph classification tasks. The results of different graph sparsity-levels are displayed in Fig.2 and Fig.3. Due to the limited space, we omit the results of weight sparsity,

¬https://github.com/yongduosui/ICPG, Apr. 2023

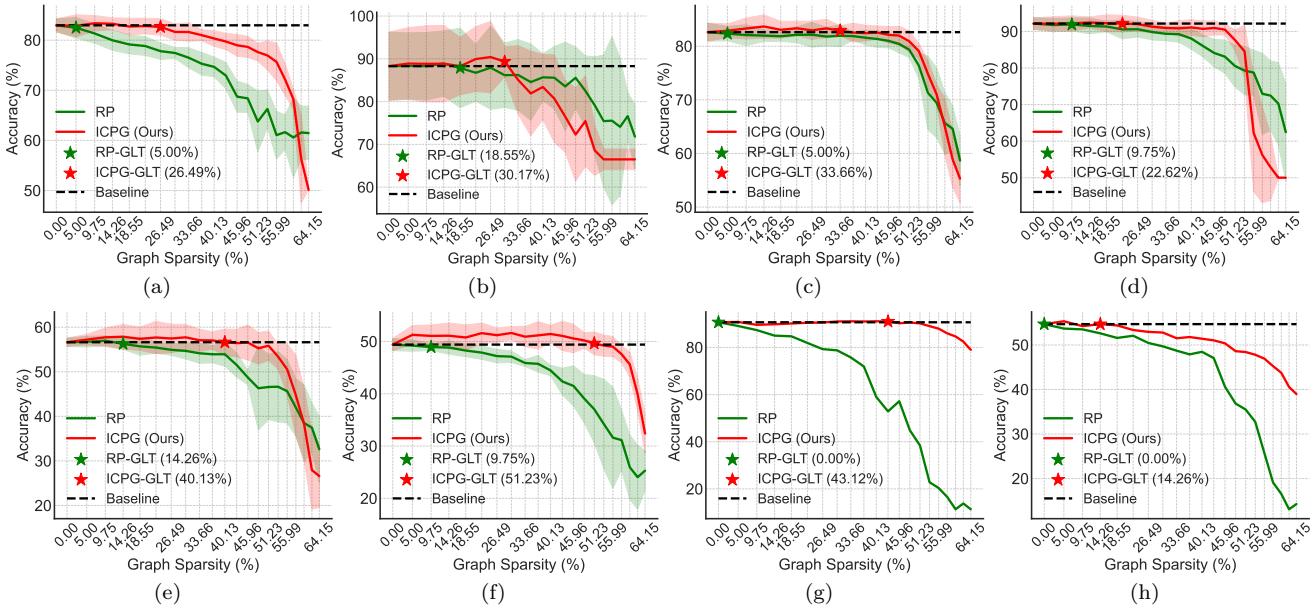


Fig.2. Graph classification performance across different graph sparsity-levels. (a) NCI1. (b) MUTAG. (c) COLLAB. (d) RED-B. (e) RED-M5K. (f) RED-M12K. (g) MNIST. (h) CIFAR-10.

which follow a similar trend. We also plot the random pruning (RP) for comparison. Stars denote extreme sparsity, which is the maximal sparsity-level without performance degradation. We make the following observations.

Observation 1. GLTs extensively exist in graph classification tasks. Utilizing the ICPG, we successfully locate the GLTs with different sparsity-levels from diverse types of graphs. For NCI1 and MUTAG, we precisely identify GLTs with extreme graph sparsity at 26.49% and 30.17%, GNN sparsity at 73.79% and 79.03%, respectively. On four social network datasets, we find the GLTs with graph sparsity of 22.62%-51.23% and GNN sparsity-level of 67.23%-95.60%. For MNIST and CIFAR-10, the GLTs are achieved with graphs sparsity of 43.13% and 14.26%, and GNN sparsity of 91.41% and 48.80%. These results show that ICPG can inductively locate the high-quality GLTs with different graph types, and demonstrate the potential of efficient training and inference with sparser graphs and lightweight GNNs without sacrificing performance.

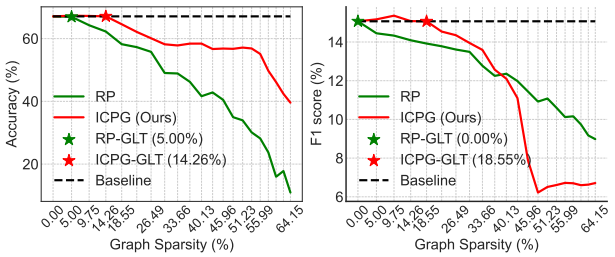
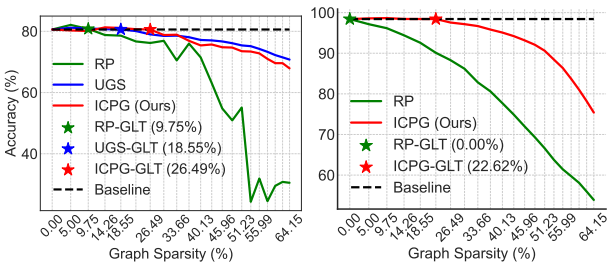


Fig. 3. Graph classification performance across different graph sparsity-levels on large-scale datasets. (a) ogbg-ppa. (b) ogbg-code2.

Observation 2. AutoMasker has good generalization ability. The mainstream graph sparsification techniques [7, 17, 19] cannot inductively prune unseen graphs. However, the AutoMasker can flexibly overcome this challenge. Compared with RP, ICPG can find more sparse subgraphs and subnetworks and keep a large gap with RP. For instance, the RED-M5K and RED-M12K graphs pruned by ICPG can achieve 40.13% and 51.23% extreme graph sparsity, improving 25.87% and 41.48% compared with RP, which keeps an extremely large superiority. These indicate that AutoMasker can precisely capture more significant core-patterns from the training graphs and have a good generalization ability to predict the high-quality masks for unseen graphs.



(a) (b)

Fig. 4. Transductive and inductive node classification performance across different graph sparsity-levels. (a) Cora. (b) PPI.

Observation 3. The extreme sparsity of GLTs depends on the property of the graphs. Although ICPG achieves higher sparsity than RP on most graphs, the improvements are not obvious on a small part of the graphs, such as biochemical molecule graphs NCI1 and MUTAG. We give the following explanations. Firstly, most of the edges in these graphs are important, such as a certain edge may correspond to a crucial chemical bond, which may drastically affect the chemical properties of the molecule if pruned. Furthermore, the graph size is relatively small, which just includes a few dozen nodes and edges, therefore it is more sensitive to pruning. The study GraphCL [33] also finds a similar phenomenon with us. It states that the performance of these chemical and molecular datasets could not be improved by data augmentation. In our work, we also experimentally demonstrate the phenomenon that edges in datasets of biochemical molecules, are more important than those of social networks. For example, on the biochemical molecule datasets, ICPG can achieve an average extreme sparsity of 28.6%, while on the social network datasets, ICPG can achieve an average extreme sparsity of 36.9%.

Observation 4. AutoMasker can well tackle larger-size and larger-quantity graphs. Fig.3 demonstrates the results on the challenging OGB datasets, which consist of larger-size graphs (2266.1 edges and 243.4 nodes on average per graph for ogbg-ppa) and larger-quantity graphs (452,741 graphs for ogbg-code2). We surprisingly find that the OGB datasets are so intractable that RP can only locate 5% graph sparsity-level of GLT on the ogbg-ppa, and it is even impossible to find any sparser GLTs on the ogbg-code2. Despite this, the proposed ICPG can locate the GLTs with 14.26% and 18.55% graph sparsity-levels, 48.80% and 59.40% GNN sparsity-level on ogbg-ppa and ogbg-code2, respectively. The superior performance further verifies the generalization ability and strong scalability.

5.3 GLTs in Node Classification Tasks

Since ICPG can achieve excellent performance on diverse types and scales of graphs, we also want to explore if it can also tackle node-level tasks. To answer this question, we conduct experiments on Cora and PPI, which are commonly used in transductive and inductive node classification tasks. We also reproduce the recent work ADMM [19,22] and UGS [7] for Cora (cannot apply for inductive setting) for comparison. From the results in Fig.4, we give the following observations.

Observation 5. ICPG achieves excellent performance in node classification tasks. Firstly, for Cora, all pruning methods consistently outperform RP and keep a large gap as the sparsity-level increases. UGS just adopts simple trainable masks for edges without considering the global topological structure of the entire graph. ADMM only optimizes the adjacency matrix without considering the GNN model. ICPG overcomes these two issues, thereby predicting more high-quality masks. Hence, ICPG can locate sparser GLTs than ADMM ("21.49%) and UGS ("7.94%). Secondly, the performance of ICPG drops faster in the later stage. These phenomena also exist in several other datasets, such as ogbg-code2, RED-B and RED-M5K. From Algorithm 2, each round of ICPG will preferentially prune the model weights and graph edges with the lowest importance score, therefore those unimportant weights and edges will be removed at an early stage. Some recent studies [34–36] have also demonstrated that there exist important features in graph data, often called causal subgraphs [35] or rationales [34,36]. These

U V U#V U+V U/V

6B;X8X S2`7Q`K M+2 Q7 /Bp2`b2 :LLb QM i?2 bT `b2 ;` T?b T`mM2/ #v miQJ bF2`X U V L*AR U:ALVX U:ALVX U/V _1.@JRkE U: hVX

h #H2X9 :` T? *H bbB}+ iBQM ++m` +v UWV

. i b2i	J2i?Q/	:` T? aT `bBiv					
		yW ULQ T`mMBM;V	NXd8W	R3X88W	jjXeeW	98XNeW	
_1.@"	_S	NkXR8	NyXey	3NXd8	3eXd8	38XR8	38Xj9
	miQJ bF2`	NkXR8	NkXRe	NRXy8	NyXR8	NyXye	3NXe9
_1.@J8E	_S	8eXej	8eXjj	88X38	89X3R	89XRN	89XN8
	miQJ bF2`	8eXej	8eX3N	8eXeN	8dXyR	8eXNd	8eXyN

72 im`2b Q7i2M /2i2`KBM2 i?2 BMi`BMbB+ K T`QT2`iv Q7 i?2 /i b2ib, _1.@" ;` T? / i - bm+? b i?2 7mM+iBQM H ;`QmT bQ2 F2Ki?2 m@L KQ/2Hb mMT`mM2/ QM H `/ i - Q` bQK2 BKTQ`i Mi 2/;2 +QHH2+iBQMb BM bQ?2 T2`7Q`K M+2 Q7 ;` T? +H M2irQ`FbX[8] S2`im`#BM; Q` T`mMBM; i?2K Kv/B22` i?2 Mi bT `bBiv@H2p2H [8] M/ 2 T`QpB/2 z2+i T2`7Q`K M+2X " b2/ QM Qm` `2bmHib- A*S#12M/2bmHib `2 b?QrM BM #QH/X iQ `2KQp2 i?2 `2/mM/ Mi T `ib Q7 i?2 /i?2BMQ`H2Q2`BM; Q#b2`p iBQMbX

bi ;2- M/ i?2 `2K BMBM; T `ib `2 # bB+ HHv i?2 B:;Mm HQJ bF2` ? b #Qi? :LL@H2p2H 72 im`2b Q7 i?2 ;` T? / i X AMimBiBp2Hv- i?2 @H2p2H Bi M/ #BHBivX 6Q` :LL@ iBQM `2bmHib BM [8 X+] #b2HBQQ`MQp2 i?2 b2`Q2B`iQK [8] 6B;X:AL M/ : h +?B2p2 ` M;@ >2M+2- i?2 T2`7Q`K M+2 Q7 A*S: rBHH BMQT bQ3W@}+8 XNeW M/ R3X88W@kkXekW BM H i2` bi ;2bX M/ _1.@JRkE rBi?Qmi b +`B}+BM; T2`7Q`K

iQJ bF2` HbQ QmiT2`7Q`Kb _S M/ F22Tb h?2b2 `2bmHib /2KQMbi` i2 i? i miQJ bF2`

8X9 h` Mb72` #BHBiv Q7 i?2 miQJ bF2`

q2 +QMbB/2` irQ T2`bT2+iBp2b iQ p2`B7v i?2 i`Mb72` #BHBiv Q7 i?2 miQJ bF2`, :LL@H2p2H i` Mb72` #BHBiv M/ ;` T?@H2p2H i` Mb72` #BHBivX 6Q` #2iiM`@K Q/2H@ ;MQbiB+ bm#;` T? /Bb+Qp2`v- M Mb72` #BHBiv B;MB}+ Mi b2K MiB+ BM i` Mb72`@ HL KQ/2H bT`b2 `2bmHi r2 i` Mb72` i?2 `2bmHib Q7 i?2 ` M;2 i` Mb72``2/ iQ /Bp2`b2 i` Mb72`i?2 bT `b2 ;` T?b T`mM2/ #v miQJ bF2` iQ Qi?2` :LL KQ/2Hb i?2 T2`7Q`K M+2 Q7 /2;` / iBQMX b Qi?2` irQ TQTmH ` :LL KQ/2Hb, h AL H2p2H i` Mb72` #BHBiv Q7 b2H2+iBQM +QKT `2 6`QK ;` T?b pB2r- r2 }`bi T`2@i` BM i?2 miQJ bF2` M/ QK T`mMBM; /2+`2 b2b b i? QM i?2 H `;2`@b+ H2 bQ+B H M2irQ`F /i b2i _1.@JRkE M/ i?2 bKMX 6Q` _1.@" M/ _1.@J8E i` Mb72` i?2 r2HH@i` BM2/ miQJ bF2` iQ T`mMBM; BM i?2 iQT Mm K#2` BMi@Q7 Qi?2` H BM+`2 b2b 7`QK y iQ 8

h #H2 8: ` T? *H bbB}+ iBQM ++m` +v UWV QM L*AR- *PGG " M/ _1.@J8E . i b2ib +`Qbb .Bz2`2Mi

J2i?Q/	L*AR			*PGG "			_1.@J8E		
	kkXekW	jjXeeW	9yXRjW	kkXekW	jjXeeW	9yXRjW	kkXekW	jjXeeW	
:` T?a :1 [Re]	deXek	dkXNd	dRXkd	djXNj	eNXy9	e3X8y	9dXe8	99Xk8	jeX3j
.`QT1/;2 [R3]	3kXk9	3RX9y	3yXR9	3kXRe	3RXNe	3RX8k	8yXjd	9eX38	98Xj8
L2m` HaT [jj`] b2	3RX9j	3yXj9	dNX3j	3RXej	ddXde	d8Xj8	8kX3k	8RX8e	9NXN8
A*S: UQm`bV	3kX3k	3RXej	3yXj9	3jXj9	3kXNy	3kX99	8dXeN	8dXyd	8eXe
AKT`Qp2K2Mi UQm`bV	"yX8W	"yXkjW	"yXkyW	"RXR3W	"yXN9W	"yXNkW	"9X3dW	"8X8RW	"eXe3W

h #H2 e AM72`2M+2 J *b *QKT `BbQMb

J2i?Q/	Jlh :	L*AR	*PGG "	_1.@"	_1.@8E	_1.@RkE	Q;#;@+Q/2k	Q;#;@TT
" b2HBM2	kjX8jJ	3j9XNdJ	j998X9jJ	9dkjXeyJ	RjeeRXeeJ	k9jeeXReJ	RjNdXN8:	8
A*S: UQm`bV	8XyNJ	kkjXdeJ	RRyjXyeJ	R83jXNdJ	R839XkjJ	R8jjX8dJ	edkXNR:	k
_2/m+iBQM UQm`bV	#md3XjeW	# djXkyW	# edXN3W	# eeX9dW	# 33X9yW	# NjXdRW	# 8RX3eW	# 9NX93W	

+m` +v /2+`2 b2b #v dXjNW M/ kXNdW 8X8 `2bT2+iBp2Hv X M/ AM72`2M+2 J *bX q?BH2 miQJ bF2`+ M +?B2p2 +QMbBbi2Mi BKT`Qp2K2Mi rBi?BM HH bT `bBiv@H2p2HbX 6m`i?2`KQ`2- i?2 :LL KQ/2H Q7 bQK2 Qi?2` i2 i?2 i` BM2/ rBi? KQ`2 bT `b2 ;` T?b 2p2M QmiT2`7Q`Kb i?2 :LL i` BM2/ rBi? i?2 Q`B;BM H /2Mb2 ;` T?b- bm+? b _1.@" iNXd8W M/ _1.@J8E iNXd8W@9X8N eWXAr /2KQMbi` i2b i? i miQJ bF2` + M r2HH i /Mb72` i?2 FMQrH2/;2 7`QK H `;2@b+ H2 mTbi`2 K i bF b iQ bK HH@b+ H2 /QrMbi`2 K i bFb M/ +?B2p2 /Q K#H2@rBM;rBi? HQr2` +QKTmi iBQM H +Qbi M/ #2ii2` T2`7Q`K M+2X 6Q` 2t KTH2- miQJ bF2` + M H2 `M KQ/2H@;MQbiB+ ;2M@ 2` H- M/ bB;MB}+ Mi bT `b2 bm#;` T? b i?`Qm;? i?2 ;` T?b- bQ i? i Bi ? b Qmibi M/BM; :LL@H2p2H T2`7Q`K M+2 Q7 i?2 ;` T?@H2p2H i` Mb72` #BHBivX

SQbi@i` BMBM; AM72`2M+2 J *bX S2 `7Q`K M+2 Q7 i?2 bQHmiBQMb i` BM2/ rBi? 2MiB`2 Q7 A*S: r2 p HB/ i2 i?2 T2`7Q`K M+2 Q7 i?2 KQ/2H i?2 2+B2M+v Q7 A*S:- r2 p HB/ i2 i?2 T2`7 i?2 `2bmHi ` Gb; q2 /QTi :` T?a :1 [kR] .`QT1/;2 [R3] M/ L2m` H aT `bB+2 [Rd] b b2HBM2b +? + M +?B2p2 ;` T? bT `b +iBQM BM /m+iBp2 HvX 6Q` 7 B` +QKT `BbQM- r2 T2`7Q`K i?2 b K2 HH@b2iiBM; bT `bB7B+ iBQM ;` T?b- Hb+ iBQM `2`2/ iQ b K HH@b+ H2 :LL :` T?a :1 Ub KTHBM; `2TQ`i 2/ M/ L2m` H aT `bB+2 bT `bB7B2b ` M/QK H- H2p2H@#v@H2p2H- M/ i?2 ;` T?b iQ +QMi BM ii2MiBQM- iQ bQK2 2ti2Mi- +QKT `2 rBi? A*S:X LQi 2 i? i 2 +? ;` T? T`QT ;2b i?2 ?B;?2bi bmT2`BQ`Biv Q7 A*S:X AM L2m` H aT `bB+2 Q7 U ;` T? T`QT ;2 M/ [R] - r2 i` BM- i?2 bT `bBiv@H2p2H iQ i?2 BM72`2M+2 J *b 2p Hm iBM; i?2 +QKTmi iBQM H +QbiX q2 `2 i`2K2 BM72`2M+2 J *b- r?B+? Bb i?2 KBMBK rBi?Qmi T2`7Q`K M+2 /2;` / iBQMX h?2 `2 b?QrM BM h #H2 eX *QKT `2/ rBi? i?2 7mHH # b2HBM2 K2i?Q/ + M bB;MB}+ MiHv `2/m+2 i?2 +QKTmi iBQM #v #Qmi 8RX3eW@NjXdRW 7`QK bK HH@b+ H2 h :V iQ H `;2@b+ H2 U2X;X- Q;#;@+Q/2kV / Qmi b +`B}+BM; T2`7Q`K M+2X h?2b2 `2bmHi i?2 T` +iB+ #BHBiv Q7 A*S:X

U V U#V

6B;XeX U V h?2 +QKT `BbQM Q7 /Bz2`2Mi 2M+Q/2`b M/ miQJ bF2` QM _1.@J8E / i b2iX U#V h?2 +QKT `BbQM Q7 2 +? +QKTQM2Mi BM A*S: QM SSA / i b2iX

8Xe #H iBQM aim/v

1M+Q/2` L2irQ`F bF2` Bb /2bB;M2/ QM :LL@# b2/ 2M+Q/2`- r?B+? H2 /b iQ ;HQ# H mM/2`@ bi M/BM; Q7 2 +? 2/;2 7`QK i?2 2MiB`2 ;` T?bX h?2 2@ 7Q`2- r2 2ti2MbBp2Hv BMp2biB; i2 i?2 BKT +i Q7 /Bp2`b2 2M+Q/2`b- bm+? b :LL@# b2/ Q` JGS@# b2/ 2M+Q/2`bX q2 + M Q#b2`p2 i?2 `2bmHib 7`QK h #H2 Xj i? i HH i?2 :LL@# b2/ 2M+Q/2`b- miQJ bF2` + M +?B2p2 ;QQ/ T2`7Q`K M+2, 98XNeW 2ti`2K2 bT `bBiv 7Q` :*L M/ 8RXkjW 7Q` :AL M/ : h- r?BH2 JGS@# b2/ 2M@ +Q/2` QMHv +?B2p2b jjXeeW 2ti`2K2 bT `bBivX Ai BM@ /B+ i2b i? i i?2 K2bb ;2@T bbBM; b+?2K2 Q7 i?2 :LL 2M+Q/2` M im` HHv +QMbB/2`b i?2 ;` T? bi`m+im`2 7`QK ;HQ# H pB2r- r?BH2 i?2 JGS@# b2/ 2M+Q/2` /Q2b MQiX

*Q@bT `bB}+ iBQM Xi?2 2z2+iBp2M2bb Q7 2 +? +QKTQM2Mi BM A*S:- r2 TTHv i?2K iQ i?2 ;` T?b M/ KQ/2H BM/2T2M/2MiHvX q2 2tTHQ`2 K bF@# b2/ S`mMBM; 7Q` :` T?b US:V- K ;MBim/2@# b2/ S`mMBM; 7Q` JQ/2H USJV- _ M/QK S`mMBM; QMHv 7Q` :` T?b U_S:V- QMHv 7Q` JQ/2Hb U_SJV- #Qi? Q7 HH U_SV- _ M/QK S`mMBM; 7Q` :` T?b rBi? K ;MBim/2@# b2/ S`mMBM; 7Q` JQ/2H U_S:@SJVX h?2 `2bmHib `2 b?QrM BM h #H2 kX q2 + M }M/ i? i, S: + M HbQ }M/ i?2 K i+?BM; bm#M2irQ`Fb i R9XkeW bT `bBiv- r?B+? Bb +QM}`K2/ `2bB? i?2 ;Q H Q7 BM i?2 +QKTmi2` pBbBQM }2H/X A*S: bB;MB}+ MiHv QmiT2`@ 7Q`Kb _S M/ _S:@SJ- M/ i?2 ; T ;` /m HHv r?2M b i?2 bT `bBiv BM+`2 b2bX q2 HbQ Q#b2`p2 i? i A*S: Bb 2p2M #2ii2` i? M S:@SJ- M/ r2 K F2 i?2 7QHHQrBM; 2tTH M iBQMbX

URV b 7Q` S:- rBi? i?2 bT `bBiv@H2p2H ;` /m HHv BM@ +`2 bBM;- i?2 ;` T?b HbQ #2+QK2 KQ`2 bT `b2- r?B+? K F2b i` BM i?2 Qp2`@T ` K2i2`Bx2/ :LL KQ/2H 2 bB2` M/ ;` T?b- Bi K v + mb2 Qp2`@}iiBM;X

UkV aHB;?iHv T`mMBM; i?2 Qp2`@T ` K2i2`Bx2/ :LL b# i?`Qm;? SJ + M #2 `2; `/2/ b FBM/ Q7 ;` /m H K` b2@

iBQM- r?B+? rBHH BKT`Qp2 i?2 T2`7Q`K M+2 bBbi2Mi rBi? Gh> `i?2`- i?2 `2;mH `Bx2/ :LL + //BiBQM HHv T`QpB/2 miQJ bF2` rBi? KQ`2 T2`pBbBQM bB;M Hb 7`QK i?2 ;` /B2Mi BM # iQ K F2 rBb2 /2+BbBQMbX AM bmKK `v- i?2b ;2bi i?2 bB;MB}+ M+2 Q7 +Q@i` BMBM; i?2 :LL- M/ +Q@bT `bB7vBM; i?2 BMTmi ;` T?b iQ +?B2p2 #2ii2` T2`7Q`K M+2X

6B;XdX oBbm HBx iBQM Q7 i?2 bm#;` T?b 2ti` +i2/ #v miQJ bF2` 7`QK *Q` M/ *A6 _@Ry bmT2`TBt2H ;` T?bX P`B;BM H ;2b M/ Q`B;BM H ;` T?b `2 /BbTH v2/ QM i?2 }`bi M/ +Q`BM@ bm#M2irQ`F r@ bT `bBiv@H2p2H Q7 _S M/ A*S: Bb e9X

8Xd oBbm HBx iBQM

hQ pBbm HHv bmQr i?2 2z2+iBp2M2bb Q7 i?2 BMTmi ;` T?b- r2 pBbm HBx2 i?2 bm#;` T?bX bB;MB}+ Mi bm#;` T?b ;` T?b- r?B+? bT `bBiv@H2p2H Q7 e9XR8W bT `bBiv@H2p2H 7`QK i?2 M/ *A6 _@Ry bmT2`TBt2H / i b2ibX 6Q` #2ii2` pBbm HBx iBQM- r2 HbQ THQi i?2 Q`B;BM H BK ;2b- Q`B;BM H ;` T?b M/QK T`mMBM; ;` T?b- r?B+? `2X/2TB+i2/ BM 6B;X d #v 7QHHQrBM; }M/BM;bX

URV 6Q` JLAah M/ *A6 _@Ry- i?2 2/;2b #2ir22M MQ/2b BMbB/2 i?2 QM i?2 /B;Bi Hb M/ Q#D2+ib / `F #Hm2 MQ/2bV b?QmH/ #2 /2Mb2`- r?B+? Bb +QMbBbi2Mi rBi? i?2 Gh> X bm#;` T?b `2 #HQ+F@HBF2 b#}+ iBQM i bFbX _S 2p2MHv `2KQp2b i?2 bmT2`TBt2Hb rBi?Qmi +QMbB/2`BM; i?2 `2@ + MMQi `2i BM i?2 T`BK `v ;` T? bi`m+im`2b rBi?Qmi +Q

BKTQ`i Mi `272`2M+2- r?B+? K F2b i?2 7Q`2M2`b;` T?b
/2bi`Qv2/ M/ b2`BQmbHv /2i2`BQ` i2b i?2 T2`7Q`K M+2X
A*S: miBHBx2b miQJ bF2` iQ H2 `M i?2 bB;MB}+ M+2 QM
2 +?2/;2 7`QK ;HQ# H pB2r M/+ M T`2+Bb2Hv T`mM2 2@
/mM/ Mi2/;2bX bi?2 JLAah A*S:;` T?- i?2 T`mM2/
2/;2b `2 K BMHv HQ+ i2/ QM MQM@/B;B i H TB t2Hb b- bmi
i?2 mTT2`@H27i- HQr2`@`B;?i +Q`M2`b M/ i?2 +2Mi2` T `iQ7
i?2 MmK#2` y M/ i?2 HQr2`@H27i +Q`M2` Q7 i?2 MmK#2` 3-
r?B H2 i?2 `2K BMBM; 2/;2b Q` MQ/2b `2 K BMHv HQ+ i2/
QM /B;Bi H TBt2HbX h?2b2 T ii2`Mb 7m`i?2` /2KQMbi` i2
i?2 ` iBQM HBiv M/ 2tTH BM #BHBiv Q7 A*S:X

e *QM+HmbBQMb

AM i?Bb rQ`F- r2 2M/Qr2/ i?2 ;` T? HQii2`v iB+F2ib
rBi? BM/m+iBp2 T`mMBM; + T +BivX q2 T`QTQb2/ bBKTH2
#mi 2z2+iBp2 T`mMBM; 7` K2rQ`F A*S:- iQ +Q@bT `bB7v
i?2 BMTmi ;` T?b M/ :LL KQ/2HX Pm` +Q`2 B/2 Bb iQ
iBQM- i?2 miQJ bF2`- H2p2` ;2b ;HQ# H +QKT`2?2MbBQM
Q7 2/;2 bB;MB}+ M+2 # b2/ QM i?2 2MiB`2 ;` T?ö iQTQ@
HQ;B+ H bi`m+im`2X h?Bb 2Mbm`2b i?2 ;2M2` HBx#BHBiv Q7
;` T?K bFb- 2t?B#BiBM; bi`QM; ;2M2` HBx iBQM + T #BH@
Biv BM BM/m+iBp2 H2 `MBM; +QMi2tibX h?2 Qm`;? 2ti2MbBp2
2tT2`BK2Mib +`Qbb p `BQmb ;` T? ivT2b- bF2H2iQMb- M/
BM; b2iiBM;b- M/i bFb- r2 +QMbBbi2MiHv 2bi #HBb?2/ i?i
A*S: + M 2z2+iBp2Hv ++QKTHBb? ?B;? bT `bBiv rBi?BM
#Qi? ;` T? / i M/ :LL KQ/2HbX h?Bb +QKT2HHBM;
2pB/2M+2 mM/2`b+Q`2b A*S:öb TQi2MiB H iQ 2z2+iBp2Hv QT@
iBKBx2 i?2 2{+B2M+v Q7 :LL KQ/2HbX
AM 7mim`2 rQ`F- r2 BMi2M/ iQ `2}M2 A*S:öb KmHiB@
`QmM/ Bi2` iBp2 T`mMBM; T ` /B;K- BMp2biB; iBM; K2i?@
Q/b iQ 2M? M+2 T`mMBM; 2{+B2M+vX h?Bb /p M+2K2Mi
+QmH/ bB;MB}+ MiHv `2/m+2 +QKTmi iBQM H +Qbib 7Q` BM;
i` BMBM;- T pBM; i?2 r v 7Q` KQ`2 `2bQm`+2@2{+B2Mi
:LL KQ/2HbX

(R)EBT7- hX EX- q2HHBM;- JX a2KB@bmT2`pBb2/ +H bb
b`T?B+ iBQM rBi? ;` T? +QMpQHmiBQM H M2irQ`FbX AM AMi2`M @
iBQM H *QM72`2M+2 QM G2 `MBM; _2T`2b2Mi iBQMb kyRdX
(k)o2HBÍFQpBÉ- SX- *m+m`mHH- :X- * b MQp - X- _
GBQ- SX- M/ "2M;BQ- uX :` T? ii2MiBQM M2irQ`FbX AM
AMi2`M iBQM H *QM72`2M+2 QM G2`MBM; _2T`2b2Mi iBQMb
kyR3X
(j)w?Qm- CX- *mB- :X- w? M;- wX- u M;- *X- GBm- wX- q
*X- amM- JX :` T? M2m` H M2irQ`Fb, `2pB2r Q7 K2i?Q/b
M/ TTHB+ iBQMbX *2T`BMi `sBp,R3Rk-yXy39j9
?iiTb,ff `tBpXQ`;f #bfR3RkXy39j9- P+i kykRX
(9)GB- :X- sBQM;- *X- h? #2i- X- :? M2K- "X .22T;+M,
2`;+M;, HH vQm M22/ iQ i` BM /22T2` ;+MbX
`sBp,kyyeXydddjN- ?iiTb,ff `tBpXQ`;f #bfkyyeXyddjN- CmM
kykyX
(8)GB- :X- JCHH2`- JX- ZB M- :X- S2`2x- AX *X .X- >
b?Qm`- X- h? #2i- X EX- :? M2K- "X .22T;+M/ +QMi@
J+ BM2;, +M2 ;Q b /22T b :+MbX AM
QM S ii2`M MvbBb M/ J +?BM2 AMi2HHB;2M+2
RyXRRyNfhS JAXkykRXjyd9y8dX
(e)>m- qX- 62v- JX- wBiMBF- JX- .QM;- uX- _2M- >X- GBm- "X- * i2bbB- JX- Gp2b+Qp2+- CX PT2M ;` T? #2M+?K `F, . i b2ib 7Q` K +?BM2 H2 `MBM; QM ;` T?bX AM .p M+2b BM L2m` H AM7Q`K iBQM S`Q+2bbBM; avbi2Kb kyky- TTX kkR3@kkRjjX
(d)*?2M- hX- amM- uX- a?B- uX- q? M;- X- q M;- wX
);` T? HQii2`v iB+F2i ?vTQi?2bBb 7Q` ;` T? M2m` H M2irQ`FbX AM S`Q+X i?2 j3i? AMi2`M iBQM H *QM72`2M+2 QM J +?BM2 G2 `MBM;- ii?2 JG_- TTX R3N8@RdyeX
(3)6` MF2H2- CX- *`#BM- JX h?2 HQii2`v iB+F2i ?vTQi?2bBb, 7BM/BM; bT `b2- i` BM #H2 M2m` H M2irQ`FbX AM AMi2`M iBQM H *QM72`2M+2 QM G2 `MBM; _2T`2b2Mi iBQMb kyRNX
(N)am- qX- >m- qX- G2bFQp2+- CX- C2;2HF - aX >Qr TQr2`7mH `2 ;` T? M2m` H M2irQ`Fb\ AM AMi2`M iBQM H *QM72`2M+2 QM G2 `MBM; _2T`2b2Mi iBQMb kyRNX
(Ry)uBM;- wX- uQm- CX- JQ``Bb- *X- _2M- sX- > KBH@ iQM- qX- G2bFQp2+- CX >B2` `+?B+ H ;` T? `2T`2b2Mi iBQM H2 `MBM; rBi? /Bz2`2MiB #H2 TQQHBM;X AM .p M+2b BM L2m` H AM7Q`K iBQM S`Q+2bbBM; avbi2Kb kyR3- TTX93yy 93RyX
(RR)*?2M- hX- 6 M- w- *? M;- aX- GBm- aX- w? M;- JX- q M;- wX h?2 HQii2`v iB+F2i ?vTQi?2bBb 7Q` T`2@i` BM2/ "1_h M2irQ`FbX AM .p M+2b BM L2m` H AM7Q`K iBQM S`Q+2bbBM; avbi2Kb kykyX
(Rk)X*? ?- J- 1{+B2Mi QM2@bFBMhM;- G2 `MBM; ;` T? M2m` H M2irQ`Fb 7Q` *QKTmi2` oBbBQM S ii2`M _2+Q;MBiBQM- kykR- TTX Rejye@RejReX

(Rk) J - >X- *?2M- hX- >m- hX@EX- uQm- *X- sB2k9QXB- qCM- w?XM;- hX- hB M- >X- CBM- aX- 6 `/ /- JX- w aT2M/BM; uQm` qBMMBM; GQii2`v "2ii2` 7i2` a;+BMM; AiXT? bT `bB}2` # b2/ QM ;` T? +QMpQHmiBQ `sB-p #bfkRyRXyjk88X ?iiTb,ff `tBpXQ`;f #bfkRyRXyjk88X +@ bB *QM72`2M+2 QM EMQrH2/;2 .Bb P+i kykRX M/ . i JBMBM;- kyky- TTX kd8 k3dX

(Rj) GBm- wX- amM- JX- w?Qm- hX- >m M;- :X- ...`2HH- hX- 2@ i?BMFBM; i?2 p Hm2 Q7 M2irQ`F +X m?2BdM?X AM@ i2`M iBQM H *QM72`2M+2 QM G2 `MBMT;_2T`2b2Mi iBQMb kyRNX

(k8) ?2M- CX- J- hX- sB Q- *X 6 bi;+M, 6 bi H2 `MBM ;` T? +QMpQHmiBQM H M2irQ`Fb pB BKTQ`i M+2 b S`Q+X i?2 ei? AMi2`M iBQM H *QM72`2M+2 QM G2 ` b2Mi iBQMb BH kyR3X

(R9) | M;- *X- w? M;- :X- :`Qbb2- _X SB+FBM; rBMMBM; iB+F2ib #27Q`2 i` BMBM; #v T`2b2`pBM;3i? AM2Mi2 MQ@X AM iBQM H *QM72`2M+2 QM G2 `MBM;T_2T`2b2Mi iBQMb

(ke) KBHiQM- qX- uBM;- wX- G2bFQp2+- CX AM/m+iBp iBQM H2 `MBM; QM H `/p M+2b BM M2m` H BM7Q`@ K iBQM T`Q+2bbBM;2b2 kyRd- TTX Ryk9 Ryj9X

(R8) a p `2b2- SX- aBHp - >X- J B`2- JX qBMMBM; i?2 HQii2`v rBi? +QMiBMmQmb bT `bB}+ iBQM AM2 b BM L2m` H AM@ 7Q`K iBQM S`Q+2bbBM; avbi2Kb j- 2@KT@Q+22/BM;b #2` kyky- TTX RRj3y@RRjNyX

(kd) uQm- >X- Gm- wX- w?Qm- wX- 6m- uX- GBM- uX #B`/ ;+Mb, :` T?@M2irQ`F +Q@QTiBKBx iBQM iQr 2{+B2Mi ;+M i` BMBM; M/ BM72`2M+2 pB /` rBM #B`/ HQii2`v iB+F2ib bXT`2T`BMi `sBp,kRyjXyydN9 ?iiTb,ff `tBpXQ`;f #bfkRyjXyydN9- .2+ kykRX

(Re) Qm/B; `B- 1X- a H K MQb- LX- S T ;2Q`;BQm- hX- u M@ M FQm/ FBb- 1X CX _ MF /2;`22, M 2{+B2Mi H@ ;Q`Bi?K 7Q` ;` T? b KTHBM; 1tT *JAMi2`@ M iBQM H *QM72`2M+2 QM /p M+2b BM aQ+B H L2irQ`Fb M HvbBb M/ JBMBM;- kyRe- TTX Rky RkNX .PA, RyXRRyNf aPL JXkyReXdd8kkkjX

(k3) ?2M- hX- "B M- aX- amM- uX `2 TQr2`7mH ; M2m` H M2ib M2+2bb `v\ /Bbb2+iBQM QM +H bbB}+ iBQMXBp T`2T`BMi `sBp,RNy8Xy98dN ?iiTb,ff `tBpXQ`;f #bfRNy8Xy98dN- CmM kykyX

(Rd) G2bFQp2+- CX- 6 HQmibQb- *X a KTHBM; 7`QK H `;2 ;` T?bX AM S`Q+X i?2 Rki? *J aA:E.. BMi2`M iBQM H +QM72`2M+2 QM EMQrH2/;2 /Bb+Qp2`v M/ / i mMBM;- kyye- TTX ejR eje- .PA, RyXRR98fRR8y9ykXRR8y9dNX

(kN) HB`2/- TX- AX- CQb?B- *X EX- G m`2Mi- hX- "2M+?K `FBM; ;` T? M2m` H M2irQ`FbX `sBp T`2T`BMi `sBp,kyyjXyyN3k ?iiTb,ff `tBpXQ`;f #bfkyyjXyyN3k- .2+ kykkX

(R3) w2M;- >X- w?Qm- >X- a`Bp bi p- X- E MM M- oX :` T?b BMi, :` T? b KTHBM; # b2/ BM/m+iBp2 H2 `MBM; K2i?Q/X AM i?2 3i? AMi2`M iBQM H *QM72`2M+2 QM G2 `MBM; _2T`2b2MiiBQMb kykyX

(jy) JQ` Bb- *X- E`B2;2- LX JX- " mb2- 6X- E2`biBM;- EX- SX L2mK MM- JX hm/ i b2i, +QHH2+iBQM Q7 #2M+?K `F / i b2ib 7Q` H2 `MBM; rBi? ;` T?bX `sBp T`2T`BMi `sBp,kyydXy3eej ?iiTb,ff `tBpXQ`;f #bfkyydXy3eej- CmH kykyX

(RN) 0` M+2b+?B- GX- LB2T2`i- JX- SQMiBH- JX- >2- sX- G2 `MBM; /Bb+`2i2 bi`m+im`2b 7Q` ;` T? M2m` H M2irQ`FbX AM iBQM H *QM72`2M+2 QM J +?BM2 G2 `MBM;- kyRN- TTX RNdk RN3kX

(jR) +? Mi - _X- a? DB- X- aKBi?- EX- Gm++?B- X- 6 aCbbi`mMF- aX aHB+ bmT2`TBt2Hb +QKT `2/ iQ bi i2 bmT2`TBt2H K2i?Q/bX Mb +iBQMb QM S ii2`M M H@ vbBb M/ J +?BM2 AMi2HHB;2M+2- pQHX j9- MQX R kk3k- LQpX kyRkX .PA, RyXRRyNfhS JAXkyRkXRkyX

(ky) uBM;- _X- "Qm`;2QBb- .X- uQm- CX- wBiMBF- JX- G2bFQp2+- CX :LL1tTH BM2`, :2M2` iBM; 1tTH M iBQMb 7Q` :` T? L2m` H L2irQ`FbXBp T`2T`BMi `sBp,RNyjXyj3N9 ?iiTb,ff `tBpXQ`;f #bfRNyjXyj3N9- LQp kyRNX

(jk) EMv x2p- "X- h vHQ`- :X qX- K2`- JX _X lM/2`bi M/BM; ii2MiBQM M/ ;2M2` HBx iBQM BM ;` T? M2m` H M2irQ`FbX S`Q+X /p M+2b BM L2m` H AM7Q`K iBQM S`Q+2bbBM; avbi2Kb kyRNX

(kR) 2- uX- CB- aX aT `b2 ;` T? ii2MiBQM M2irQ`FbX +iBQMb QM EMQrH2/;2 M/ . i JBMBM;- kykRX .PA,RyXRRyNfhE.1XkykRXjydkj98X

(jj) uQm- uX- *?2M- hX- amB- uX- *?2M- hX- q M;- wX- a?2M- uX ;` T? +QMi` biBp2 H2 `MBM; rBi? m;K2MiiBQMbX /p M+2b BM L2m` H AM7Q`K iBQM S`Q+2bbBM; avbi2Kb- +2K#2` kyky- TTX 83Rk@83kjX

(kk) w?2M;- *X- wQM;- "X- *?2M;- qX- aQM;- .X- LB- CX- um- qX- *?2M- >X- q M;- qX _Q#mbi ;` T? `2T`2b2Mi iBQM H2 `MBM; pB M2m` H bT `bB}+ iBQMXAM iBQM H *QM72`2M+2 QM J +?BM2 G2 `MBM;- kyky- TTXRR983 RR9e3X

(j9) uQm- uX- *?2M- hX- a?2M- uX- q M;- wX- G2bFQp2+- CX- +QMi` biBp2 H2 `MBM; pB ;` T? m;K2Mi iBQMX AM i?2 3i? AMi2`M iBQM H *QM72`2M+2 QM G2 `MBM; _2T`2b2Mi iBQMb kykkX

(kj) _QM;- uX- >m M;- qX- sm- hX- >m M;- CX .`QT1/;2, iQr `/b /22T ;` T? +QMpQHmiBQM H M2irQ`Fb QM MQ/2 +H bbB}+ iBQMX AM i?2 3i? AMi2`M iBQM H *QM72`2M+2 QM G2 `MBM; _2T`2b2Mi iBQMb kykyX