Improving Synchronization in Graph Neural Diffusion with the Kuramoto Model

Tuan Nguyen
FPT Software AI Center

Hirotada Honda
Toyo University

Takashi Sano
Toyo University

Shugo Nakamura
Toyo University

Vinh Nguyen
FPT Software AI Center

Tan M. Nguyen
National University of Singapore

Abstract

We propose the Kuramoto Graph Neural Network (KuramotoGNN), a class of continuous-depth graph neural network architectures based on the Kuramoto model. The Kuramoto model describes the synchronization behavior of non-linear coupled oscillators. Under the view of coupled oscillators, the oversmoothing phenomenon can be considered as the well-known phase synchronization phenomenon. The KuramotoGNN has two theoretical properties: (i) under the dynamics of the KuramotoGNN, the node representations do not converge to the same state even in the very deep model, which alleviates the oversmoothing issue in graph neural networks. (ii) the KuramotoGNN can provide solid theoretical results to the stability of the synchronized state. We experimentally verify the above two advantages on various graph deep learning benchmark tasks, showing a significant improvement over many existing graph neural networks.

1 INTRODUCTION

Graph neural networks have succeeded in various applications, including computational chemistry (Gilmer et al. (2017)), social networks (Fan et al. (2019)), and drug discovery (Xiong et al. (2019)). The essential idea of GNNs is to design multiple graph propagation layers to iteratively update each node representation by aggregating the node representations from their neighbors and the representation of the node itself. Despite these successes, it has been observed that the representations of the graph nodes of different classes would become indistinguishable when increasing the depth of GNNs as the result of taking a weighted average of its neighbors' features. Thus, the performance of the model is significantly dropped. This common phenomenon has been known as the *oversmoothing* issue (Oono and Suzuki (2019); Nt and Maehara (2019)).

It is crucial to improve our theoretical understanding of GNNs to alleviate the oversmoothing problem when node representations converge to the same value so that we are able to build deeper networks for learning more complex representations of the data. Following Chen et al. (2018), recent works on understanding GNNs have considered the model as a discretization scheme of dynamical systems, i.e., each propagation layer in GNNs is a discrete step of a differential equation (Chamberlain et al. (2021); Thorpe et al. (2021); Rusch et al. (2022); Xhonneux et al. (2020); Oono and Suzuki (2019)). This class of models is often called continuous-depth models. These models are more memory-efficient and can capture the dynamics of hidden layers effectively (Chen et al. (2018)).

Previous works used diffusion-based methods to propose a new propagation scheme for GNNs (Chamberlain et al. (2021); Thorpe et al. (2021); Xhonneux et al. (2020)). Different from these works, we use the *Kuramoto model* (Kuramoto (1975)), which describes the dynamical behavior of non-linear coupled oscillators, as a backbone framework to explain the GNNs and to alleviate the oversmoothing problem. In particular, we realize the oversmoothing phenomenon is identical to the *phase synchronization* in the background of coupled oscillators, where all oscillators in the network start spontaneously rotating with a common frequency and phase. In fact, phase synchronization is common in nature, from the spiking of neurons (Varela et al. (2001)) and the flocking behavior in birds (Acebrón et al. (2005)) to the entrainment laser arrays (Strogatz and Mirollo (1993)) and seasonal diseases (He and Stone (2003)).

Proceedings of the 26th International Conference on Artificial Intelligence and Statistics (AISTATS) 2023, TBD, USA. PMLR: Volume XXX.

In the network of coupled oscillators, each oscillator has both intrinsic and extrinsic factors. When these intrinsic and extrinsic factors satisfy certain conditions, the synchrony emerges (Kuramoto (1975); Strogatz (2000); Acebrón et al. (2005)). In this paper, we try to answer the question: *Can the GNNs be trained to satisfy the conditions giving rise to synchronization or de-synchronization?* Our contributions are three-fold:

- We propose the KuramotoGNN, a new propagation scheme for GNNs based on the Kuramoto model.
- We formulate the oversmoothing problem in GNNs in terms of the collective synchronization phenomenon. Then, we derive a condition to avoid oversmoothing.
- We provide an extensive empirical evaluation of the KuramotoGNN on a wide variety of graph learning tasks, demonstrating that the KuramotoGNN achieves competitive performance compared to other variants of GNNs.

2 PRELIMINARIES

2.1 Graph Neural Diffusion

Graph Neural Diffusion is a continuous-depth architecture for graph based on the diffusion process (Chamberlain et al., 2021). Given a graph $G = (V, E)$ where E is the edge set, $V = [v_1, ..., v_n] \in R^{n\times f}$ is the set of n vertices and each vertex has f features. The model is governed by the following diffusion equation:

$$\frac{\mathrm{d}X(t)}{\mathrm{d}t} = \mathrm{div}(G(X(t), t) \odot \nabla X(t)) \tag{1}$$

$$X(0) = \psi(V) \tag{2}$$

where $X(t) = ([x_1(t)]^\top, ..., [x_n(t)]^\top)^\top \in R^{n\times d}$, div is the divergence operator, ∇ is the gradient, $\odot$ is the Hadamard product, and ψ is the encoder function to the input node features. In the simplest case, when G is only dependent on the initial value, which is $X(0)$, then the equation becomes:

$$\frac{\mathrm{d}X(t)}{\mathrm{d}t} = (\hat{A} - I)X(t) \tag{3}$$

where $\hat{A}$ is an $n \times n$ matrix with same structure with weight matrix of the graph, and further more, $\hat{A}$ is right-stochastic, i.e., each row of $\hat{A}$ summing to 1. This property can be related to attention weight. In GRAND model (Chamberlain et al., 2021), to formulate the matrix $\hat{A}$, they used the multihead self attention mechanism (Vaswani et al., 2017). The scaled-dot product attention is given by:

$$A^l(X_i, X_j) = softmax(\frac{(W_K X_i)^\top W_Q X_j}{d_k}) \tag{4}$$

where W_K, W_Q are the learnable parameters, d_k is the dimension of W_K. Then, $\hat{A} = \frac{1}{h}\sum_{l=1}^{h} A^l(X)$ with h is the hyper-parameter that determined the number of heads.

With these properties, Chamberlain et al. (2021); Thorpe et al. (2021) have showed that many GNN architectures, including GAT (Velickovic et al. (2017)) and GCN (Kipf and Welling, 2016), can be understood as a discretisation of (3). Furthermore, the use of time-independent G also makes the model become lightweight and more data-efficient than conventional GNNs.

Models that are derived from (3) has been shown that can extend to larger depth than conventional GNNs while still maintaining acceptable performance (Chamberlain et al., 2021). However, in the section 3, we argue that even with this type of model, the oversmoothing can not be completely eliminated.

2.2 The Kuramoto model

Winfree (1967) showed that, under weak coupling, a dynamical behavior of interacting limit-cycle oscillators can depend only on their phases. These kinds of model are called phase-reduced model.

One of the most famous phase-reduced model is the Kuramoto model. The model describes the dynamics of a network of N phase oscillators θ_i with natural frequencies ω_i, and coupling strength κ_{ij} under the following phase equation:

$$\dot{\theta}_i = \omega_i + \sum_j \kappa_{ij} \sin(\theta_j - \theta_i) \tag{5}$$

Each oscillator i is characterized by intrinsic and extrinsic factors (Acebrón et al., 2005). Here, the internal influence is the natural frequencies ω_i, and the external influences are interactions with other oscillators in the network via the weights (or coupling) matrix $\kappa \in R^{n\times n}$. From (5), Kuramoto (1975) analyzed furthermore the synchronization behavior by using the mean-field coupling, that is $\kappa_{ij} = K/N$ where K is a constant and N is the number of nodes in the graph. Together with fully connected and equally weight graph, (5) becomes:

$$\dot{\theta}_i = \omega_i + \frac{K}{N}\sum_j \sin(\theta_j - \theta_i) \tag{6}$$

The above equation is well-known as the *Kuramoto model*. Kuramoto model has two types of states: a nonsynchronized state, in which each oscillator rotates with its natural frequency, and partially synchronized states, in which part of the oscillators rotate with the same effective frequency. Kuramoto (1975) found that strengthening the couplings provides a synchronization transition from the nonsynchronized state to the partially synchronized states, and the continuity of the transition is determined by the natural frequency distribution. To be specific, if the distribution of natural frequencies ω_i is uni-modal and symmetric, then synchronization would occur in (6) if coupling parameter K exceeds a certain threshold $K_{critical}$ that is determined by distribution of ω_i. If $K >> K_{critical}$ then we would have total synchronized state, or the *phase synchronization* in which the positions of all θ_i are identical to each other.

To investigate the synchronization behavior, or to measure the synchronization rate, Kuramoto (1975) introduced a complex-valued term called *order parameter*:

$$re^{i\phi} = \frac{1}{N}\sum_j e^{i\theta_j} \tag{7}$$

where $\phi = \frac{1}{N}\sum_j \theta_j$ is the average phase. $r(t)$ is real-value in range of $0 \leq r(t) \leq 1$ where $r(t) \approx 1$ indicates the phases are close together, or synchronization occurs more and more if $r(t)$ is large. In contrast, $r(t) \approx 0$ indicates phases are spread all over the circle.

3 KURAMOTO GRAPH NEURAL NETWORK

3.1 Model Formulation

In this work, we consider this generalized form of the Kuramoto model:

$$\dot{\theta}_i = \omega_i + \frac{K}{\sum_j a_{ij}} \sum_j a_{ij} \sin(\theta_j - \theta_i) \tag{8}$$

If the $A = [a_{ij}]$ matrix is symmetric, then the frequency of synchronization is equal to the average of the natural frequencies. However in the case of non-symmetric, synchronization frequency does not necessarily equal to the average frequency. Instead, it depends on coupling strength (El Ati and Panteley, 2012).

As stated in 2.1, the edges weight matrix $\hat{A} = [a_{ij}]$ is a right-stochastic matrix. And thus, with this condition, together with (8), we now can present the KuramotoGNN, a class of continuous-depth GNNs based on the Kuramoto model:

$$\dot{X} = \Omega + K \sum_j a_{ij} \sin(x_j - x_i) \tag{9}$$

with $\Omega = (\omega_1^\top, ..., \omega_n^\top)^\top \in R^{n\times d}$ and initial value $X(0) = \Omega = \psi(V)$. Different from many previous works that try to find the threshold $K_{critical}$ to control the dynamics of the system, we approach the problem in a data-driven oriented way. We fix K as a hyper-parameter, and optimize the ω_i through supervised learning.

To demonstrate the flexibility of KuramotoGNN, we now show that KuramotoGNN can be an extended version of GRAND. Assuming that ω_i are identical, in which $\omega_1 = ... = \omega_N$, we obtain the following equation by using the rotating frame of reference:

$$\dot{X} = K \sum_j a_{ij} \sin(x_j - x_i) \tag{10}$$

Furthermore, if we assume that $K = 1$ and roughly approximate the nonlinear sin function by using the first order of the Taylor expansion, we obtain the following equation that is identical to (3):

$$\dot{X} = \sum_j a_{ij}(x_j - x_i) \tag{11}$$

The KuramotoGNN model, in general, is nonlinear even in the case that we consider a_{ij} is time-dependent, i.e., depends on $(x_i(t), x_j(t))$, or depends only on initial value $(x_i(0), x_j(0))$. In GRAND, if $\hat{A}$ in (3) depends only on the initial value, then we have a linear version of GRAND that they called GRAND-l. Meanwhile, if $\hat{A}$ is time-dependent, we obtain the nonlinear GRAND called GRAND-nl. In this work, we consider fixed a_{ij} only, which means a_{ij} are still learnable parameters but depends only on the initial value. Following Chamberlain et al. (2021), we used multi-head scaled-dot product attention (Vaswani et al., 2017) to calculate weight strength a_{ij} between a pair of nodes (x_i, x_j).

3.2 Oversmoothing as synchronization

Oversmoothing has been observed and analyzed in conventional GNN architectures (Oono and Suzuki, 2019; Nt and Maehara, 2019). The previous reference showed that oversmoothing is a phenomenon that the node representations features converge to a fixed state exponentially and lead to indistinguishable between nodes representations. We realize that this phenomenon is very close to phase synchronization in coupled oscillators dynamics that we present now.

Definition: We say that the oversmoothing phenomenon happens if

$$\lim_{t\to\infty} |x_i(t) - x_j(t)| = \mathbf{0}, \forall i \neq j. \tag{12}$$

In case of single channel, or when $d = 1$, the above definition is exactly the same as the definition of phase synchronization in the Kuramoto model. We first show that without intrinsic features ω_i or the intrinsic features ω_i are identical in the entire system (10), oversmoothing can be understood via locally exponentially stable solution in all d channels of X. For simplicity, in here, we consider $d = 1$ and we denote $x_i = x_{i,1}$. (10) can be represented in the following form:

$$\dot{X} = -KB\sin(B^\top X) \tag{13}$$

with $B \in R^{|V|\times|E|}$ is the incidence matrix of the graph, $B_{kl} = 1$ if node k is the sink node of edge l, and $B_{kl} = -1$

if node k is the source node of edge l. $\sin(.)$ function here is the element-wise function which means $\sin(X) = [\sin(x_1), ..., \sin(x_n)]$. We also consider the following energy function U:

$$U(X) = \sum_{i,j \in E} a_{ij}(1 - \cos(x_i - x_j)) \tag{14}$$

Because $\frac{\partial U}{\partial x_i} = \sum_{i,j \in E} a_{ij} \sin(x_i - x_j)$, we can represent $\nabla U(X) = [\frac{\partial U}{\partial x_1}, ..., \frac{\partial U}{\partial x_n}]$ as following:

$$\nabla U(X) = B \sin(B^\top X) \tag{15}$$

Which leads to:

$$\dot{U} = \nabla U(X)\dot{X} = -\frac{1}{K}\dot{X}^\top \dot{X} \leq 0 \tag{16}$$

Therefore, $U(X)$ is a positive function $0 \leq U(X) \leq 1$ and also it is a non-increasing function. Following LaSalle Invariance Principle (see Khalil, 2002, Theorem 4.4), in which the theorem asserts that every solution of (13) converges to set of critical points that are the root of the right hand side of (13), or the equilibrium solutions. Looking to (11), it is easily confirmed that oversmoothing phenomena can be expressed through fixed points, in other words, $x_i = x_j, \forall i \neq j$ is a solution of (11). We have shown that in case of single channel of node representations, (11) is likely encountered the oversmoothing phenomena. For multi-channels or when $d > 1$, the same conclusion can be derived in a similar way.

About the case of non-identical oscillators, there are works that studied well the asymptotic stability of synchronized state in the Kuramoto model, we refer to such works like Jadbabaie et al. (2004); Chopra and Spong (2009) for detailed proof.

Hence, if we can find a condition or appropriate initial value for (9) to avoid the basin of attraction of phase synchronization or make sure that the phase synchronization is not a fixed point, then we can completely avoid the oversmoothing phenomenon.

We next present a theorem which offers an easy way to reduce the oversmoothing phenomenon by introducing natural frequencies ω_i.

Theorem 1 *Let $x_i(.), i = 1, \dots, N$ be a solution of (9). If there exists $i, j \in 1, \dots, N$ such that $\omega_i \neq \omega_j$ then (12) does not happen.*

Proof 1 (Proof of Theorem 1) *We prove the contrapositive. It means that if (12) happens then $\omega_i = \omega_j, \forall i, j = 1, \dots, N$.*

We prove by contradiction. We assume that (12) happens and $\omega_1 \neq \omega_2$, then we will try to reach a contradiction.

Since (12) happens, we subsitute it into (9) to have the following limits

$$\lim_{t \to \infty} (\dot{x_i}(t) - \omega_i) = \mathbf{0}, \;\; i = 1, 2. \tag{17}$$

Now for all $N \in Z^+$, we apply the mean value theorem to have

$$x_1(N+1) - x_1(N) = \dot{x_1}(a_N), a_N \in (N, N+1),$$
$$x_2(N+1) - x_2(N) = \dot{x_2}(b_N), b_N \in (N, N+1).$$

Using (17), we get

$$x_1(N+1) - x_1(N) \to \omega_1 \;\; as\ N \to \infty,$$
$$x_2(N+1) - x_2(N) \to \omega_2 \;\; as\ N \to \infty.$$

Hence

$$d_{12}(N+1) - d_{12}(N) \to \omega_1 - \omega_2 \neq 0 \;\; as\ N \to \infty,$$
$$d_{12}(t) = x_1(t) - x_2(t)$$

which is a clear contradiction to the fact that (12) happens.

We explain how our model can avoid oversmoothing with the help of Theorem 1. Since we set $\omega_i = \psi(V_i)$, where ψ is a learnable function. There is no chance that $\omega_i = \omega_j, \forall i \neq j$ unless all the input node features V_i are identical which is not possible in practical experiments.

4 EXPERIMENTAL RESULTS

We conduct experiments to compare the performance of our proposed method KuramotoGNN with GRAND and several other popular GNN architectures on node classification tasks, including GCN, GAT, and GraphSage. For all experiments, we run 100 splits for each dataset with 20 random seeds for each split, which are conducted on a server with one NVIDIA RTX 3090 graphics card.

Except for 2 factors: the coupling hyper-parameter, which belongs solely to the KuramotoGNN, the integration time, which measures the implicit depth of continuous-model were slightly changed. For other settings, we adopt from GRAND in Chamberlain et al. (2021) for KuramotoGNN include adaptive numerical differential equation solvers. We provide detailed descriptions of experimental settings in the Supplementary material.

Following Chamberlain et al. (2021), we study seven graph node classification datasets, namely CORA (McCallum et al., 2000), CiteSeer (Sen et al., 2008a), PubMed (Sen et al., 2008b), CoauthorCS (Shchur et al., 2018), the Amazon co-purchasing graphs Computer and Photo (McAuley et al., 2015).

Table 1: Statitics of 6 datasets

Dataset	Classes	Features	#Nodes	#Edges
CORA	7	1433	2485	5069
Citeseer	6	3703	2120	3679
Pubmed	3	500	19717	44324
CoauthorCS	15	6805	18333	81894
Computer	10	767	13381	245778
Photo	8	745	7487	119043

4.1 Dataset

The statistics of the datasets are summarized in Table 1.

Cora (McCallum et al. (2000)). A scientific papers citation network dataset consists of 2708 publications which are classified into one of 7 classes. The citation network consists of 5429 links, each publication is represented by a vector of 0/1-valued indicating the absence/presence of the 1433 words in a corpus.

Citeseer (Sen et al. (2008a)). Similar to Cora, Citeseer is another scientific publications network consists of 3312 publications and each publication is classified into one of 6 classes. The publication is represented by a vector of 0/1 valued that also indicating the absence or presence of the corresponding word from a dictionary of 3703 unique words.

Pubmed (Sen et al. (2008b)). The Pubmed dataset consists of 19717 scientific publications that related to diabetes, and all publications in the dataset are taken from Pubmed database. Each publication is classified into one of 3 classes. The network has 44338 links and each publication is represented by TF/IDF weighted word vector from a dictionary consists of 500 unique words.

CoauthorCS (Shchur et al. (2018)). The CoauthorCS is co-authorship graph of authors with publications related to Computer Science field. The dataset is based on the Microsoft Academic Graph from the KDD Cup 2016 challenge. In this dataset, nodes represent the authors and an edge is established if they are co-authored in a paper. Each node is classified to one of 15 classes, and each node is represented by a vector of size 6805 indicating the paper keywords for each author's papers. The network consists of 18333 nodes and 163788 edges.

Computers (McAuley et al. (2015)). Computers dataset is a segment of the Amazon co-purchase graph. In this graph, each node is classified into one of 10 classes, and each node is represented as a product. If two products are often bought together, an edge will be established. Each product is represented by a bag-of-words features vector of size 767. The dataset consists of total 13752 product and 491722 relations between two products.

Photo (McAuley et al. (2015)) Similar to Computers, Photo is another segment of Amazon co-purchase graph, the properties of nodes and edges are exactly the same with Computers. In this dataset, the network consists of 238163 edges and 7650 nodes, in which each node is classified into one of 8 classes and each node is represented by a vector size of 745.

4.2 KuramotoGNN is resilient to deep layers

To demonstrate that the model does not suffer from over-smoothing, we conducted experiments in various of depth (by chaning the integration limit T) and measure the performance in two metrics: accuracy and synchronization rate.

One thing to notice that in original implementation of GRAND, they modified (3) as following:

$$\frac{\mathrm{d}X(t)}{\mathrm{d}t} = \alpha(\hat{A} - I)X(t) + \beta X(0) \tag{18}$$

with learnable α and β parameters. Thorpe et al. (2021) has argued in their discussion that this trick is task specific, while it increases the test accuracies for some of the benchmarks (Cora, Citeseer and Pubmed), it harms the performance on most other benchmarks. In fact, (18) is quite similar to our proposed (9) if we roughly approximating the $\sin$ function as stated before.

In this experiment, we conducts both two versions of linear GRAND, with and without adding $X(0)$. For all models, we used random split method with 10 initialization, along with fixed-step solver Euler with step size 0.1 for fair comparison in computational process instead of using adaptive step size scheme which give more superior results (Chamberlain et al., 2021).

Figure 1 showed the change in accuracy of three kinds of models: **KuramotoGNN**, **GRAND-l** and **GRAND-l w/o X(0)** for various depth values $T = \{1, 4, 8, 16, 32, 64, 80, 100\}$. We can notice that without adding $X(0)$, the performance of **GRAND-l** reduces significantly along the depth, while the **KuramotoGNN** and **GRAND-l w/o X(0)** maintain the performance when increasing the depth.

Figure 2 showed the change in synchronization rate of **KuramotoGNN** and **GRAND-l** along the depths. In this figure, we plotted the order parameter (7) at terminal time $r(T)$. There is a clear gap between two models in both datasets, the synchronization degree of GRAND-l are higher than KuramotoGNN in every depth, while in KuramotoGNN, the synchronization rate reduces when the depth is increased. Specially, the synchronization rates of GRAND-l in Cora dataset are always near 1 which indicates the oversmoothing phenomenon tends to happen.

4.3 KuramotoGNN performs well with limited labeled training data

Besides helping to avoid oversmoothing and be able to train in deep layers, KuramotoGNN also can boost performance of different tasks with low-labeling rates. Table 2 compares the accuracy of fine-tuned Kuramoto with GRAND-l (following (18)) and other popular GNNs: GCN (Kipf and Welling, 2016)), GAT (Velickovic et al., 2017), GraphSAGE (Hamilton et al., 2017).

We used the fine-tuned values that are reported by Chamberlain et al. (2021) to reproduce the GRAND-l (with X(0) term) results, for other models, we use the reproduced results from Thorpe et al. (2021). We notice that with few labeled data, in most tasks KuramotoGNN is significantly more accurate than the other GNNs including GRAND-l. Only for CoauthorCS and Photo datasets, the GCN outperform both KuramotoGNN and GRAND-l on extreme limited label cases.

4.4 Effect of coupling strength K on KuramotoGNN

To further investigate the effect of hyper-parameter K using empirical results, in the following experiments, we tried different settings of $K = \{0.4, 0.6, 0.8, 1, 1.5, 2, 3, \}$ on Citeseer dataset using standard Planetoid split and on different depth $T = \{2, 4, 8\}$.

Figure 4 showed the change in performances of Kuramoto on Citeseer dataset on different settings of K. It is observed that the KuramotoGNN performs well on small values of K while for too small K, it indicates not so much change for the coupled function, and for higher K, the performances start decreasing. However, that phenomenon quite matches with the analysis of the Kuramoto model (Kuramoto, 1975; Strogatz, 2000), in which the higher the coupling strength K, the system tends to synchronize better. Furthermore, we also do not suggest putting K too high, since it will increase the NFE (Number of Function Evaluations) of the solver to obtain an accurate solution, and thus, increasing the time of training.

4.5 KuramotoGNN performs well with noisy training data

To demonstrate the robustness of KuramotoGNN, we conduct experiments of noisy Citeseer dataset where the inputs have been perturbed by Gaussian noise. To be specific, we trained and evaluated the performance in Citeseer dataset following Planetoid split.

Figure 3 demonstrated the robustness of KuramotoGNN when compared to GRAND-l model. In general, KuramotoGNN is more robust than GRAND-l, and we also noticed that when increasing the coupling strength, the KuramotoGNN becomes better.

5 RELATED WORKS

Neural ODEs. Neural ODEs (NODE) is a class of continuous-depth model for neural network based on Ordinary Differential Equations. The idea of NODE is first proposed by Chen et al. (2018), but previously, there were studies about the relation between deep learning and differential equations (Haber and Ruthotto, 2017). Mathematically, NODE is represented as following first order ODE:

$$\frac{\mathrm{d}z(t)}{\mathrm{d}t} = f(z(t), t, \theta) \tag{19}$$

where $f(z(t), t, \theta)$ is specified by a neural network and θ is it's weights. Using numerical methods, for example the Euler discretization with step size equal 1, (19) turns to vanilla residual network (He et al., 2016). Defining neural network as differential equations has an advantage in training process. Instead of saving all states of intermediate layers for back-propagation process, ODE solvers have a method called adjoint method (Pontryagin et al., 2018) that can trace back the gradient value by solving a reverse ODE, which is memory-efficient since it does not need to use the states of forward process. Following the work of Chen et al. (2018), many works have explored the use of ODEs in deep learning, like Neural CDE (Kidger et al., 2020), Neural SDE (Liu et al., 2019), augmented NODE (Dupont et al., 2019), in such tasks of image classification and time-series prediction.

GNNs. Graph Neural Networks (Kipf and Welling, 2016; Velickovic et al., 2017; Hamilton et al., 2017) are class of model that can learn the representations of graphs effectively. Generally, GNNs model have multiple propagation layers, where in each layer the representation of each node is updated according to representation features of neighbouring nodes, which are called message. Each type of model has a different type of message aggregating function, for instance, GCN (Kipf and Welling, 2016) uses first-order Chebyshev polynomial.

Although the effectiveness of GNNs, it has been shown that these kind of models are prone to oversmoothing (Oono and Suzuki, 2019; Nt and Maehara, 2019), when node representations converge to same values. Especially, Oono and Suzuki (2019) has analytically shown that deeper GCNs exponentially lose expressive power as the number of layers goes to infinity, and two nodes with equal degree in the same connected component are will have the same representation. Some of the early solutions to this phenomenon are adding residual layers, and concatenation. Although

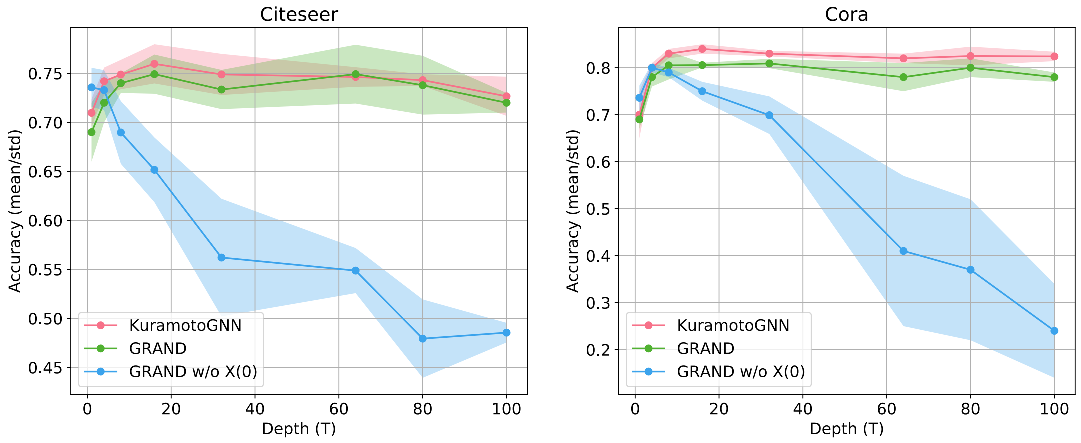

Figure 1: Change in performance at different depth (T) on Cora and Citeseer dataset.

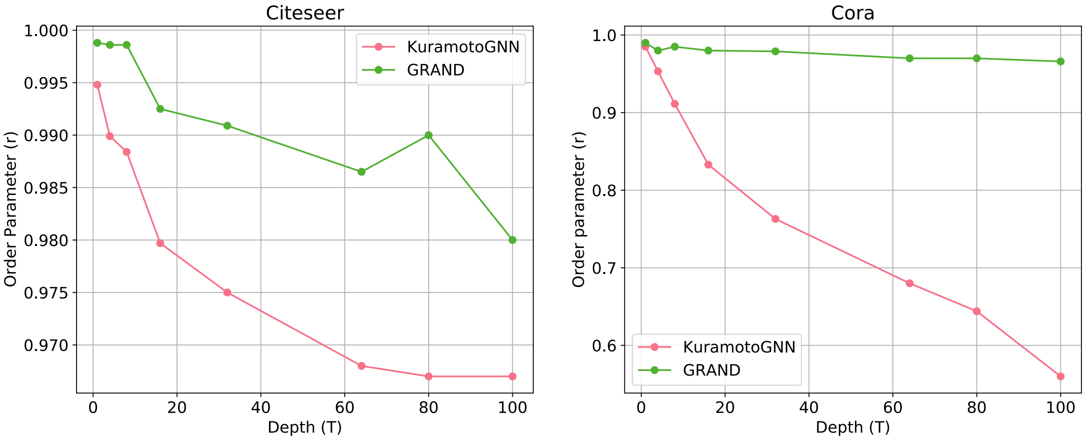

Figure 2: Change in order parameter at different depth (T) on Cora and Citeseer dataset.

the concatenating approach is effective, it does not easily scale to very deep networks due to the size of latent features increasing by numbers of layers. Following the work of Neural ODEs by Chen et al. (2018), recent works have turned conventional GNNs to continuous-depth model and this approach can solve the oversmoothing problem to some extends, for example, Thorpe et al. (2021); Xhonneux et al. (2020) added values that they called source terms to the linear ODE to change the converge point of the equation, Rusch et al. (2022) modeled the GNNs as second-order oscillators PDEs with damping term and analyzed the dynamic behavior of the model to avoid oversmoothing, and Chamberlain et al. (2021) used heat diffusion-type PDEs to design GNN and to some extend, slowed the process of oversmoothing.

Synchronization. Synchronization of complex network is observed in biological, chemical, physical, and social systems. One typical example is synchronous flashing of fireflies. Initially they flash incoherently, but after a short period of time the whole swarm is flashing in unison. All these dynamics can be studied under the view of coupled oscillator network. A coupled oscillator network is characterized by a population of oscillators and a graph which describing the interaction between oscillators. The oscillator is simple to define, yet, bring to us a rich dynamic behavior (Dörfler and Bullo, 2014). Huygen was the one of the first person studied the synchronization between coupled pendulum clocks. After that, Winfree (1967) discovered a phase transition from incoherent state to synchrony (or coherent) state, but his mathematical model was too difficult to be analytical tractable. And then, based on these works, Kuramoto (1975) studied the synchronization behavior in dynamic model for fully interaction network with uniform weights. His work nowadays known as the *Kuramoto model*.

There are studies showed that the Kuramoto model is appli-

Table 2: Mean and std of classification accuracy of KuramotoGNN and other GNNs with different number of labeled data per class (#per class) on six benchmark graph node classification tasks. The highest accuracy is highlighted in bold for each number of labeled data per class. (Unit: %)

Model	#per class	CORA	Citeseer	Pubmed	Computers	CoauthorCS	Photo
KuramotoGNN	1	**63.48±7.2**	**62.06±4.55**	**65.93±3.65**	**62.26±7.73**	60.48±2.7	80.18±1.8
	2	**71.17±5.0**	**66.85±6.72**	**72.62±3.15**	76.24±2.72	75.89±0.73	82.67±0.8
	5	**79.11±0.91**	**72.42±2.0**	**76.43±1.73**	81.43±0.78	**87.22±0.99**	**89.35±0.29**
	10	**83.53±1.36**	**74.27±1.5**	76.86±2.17	**83.84±0.54**	**90.49±0.28**	**91.35±0.1**
	20	**85.18±1.3**	**76.01±1.4**	**80.15±0.3**	**84.6±0.59**	**92.35±0.2**	**93.99±0.17**
GRAND-l with $X(0)$	1	54.14±11.0	50.58±17.3	55.47±12.5	47.96±1.3	58.1±4.6	76.89±2.25
	2	68.56±9.1	57.65±13.2	69.71±7.01	75.47±1.7	75.2±4.2	80.54±2.3
	5	77.52±3.1	67.48±4.2	70.17±4.52	81.23±0.6	85.27±2.1	88.58±1.7
	10	81.9±2.4	71.7±7.3	**77.37±2.31**	82.71±1.5	87.6±1.8	90.95±0.6
	20	82.46±1.64	73.4±5.05	78.8±1.63	84.27±0.6	91.24±0.4	93.6±0.4
GCN	1	47.72±15.33	48.94±10.24	58.61±12.83	49.46±1.65	**65.22±2.25**	**82.94±2.17**
	2	60.85±14.01	58.06±9.76	60.45±16.20	**76.90±1.49**	**83.61±1.49**	**83.61±0.71**
	5	73.86±7.97	67.24±4.19	68.69±7.93	**82.47±0.97**	86.66±0.43	88.86±1.56
	10	78.82±5.38	72.18±3.48	72.59±3.19	82.53±0.74	88.60±0.50	90.41±0.35
	20	82.07±2.03	74.21±2.90	76.89±3.27	82.94±1.54	91.09±0.35	91.95±0.11
GAT	1	47.86±15.38	50.31±14.27	58.84±12.81	37.14±7.87	51.13±5.24	73.58±8.15
	2	58.30±13.55	55.55±9.19	60.24±14.44	65.07±8.86	63.12±6.09	76.89±4.89
	5	71.04±5.74	67.37±5.08	68.54±5.75	71.43±7.34	71.65±4.56	83.01±3.64
	10	76.31±4.87	71.35±4.92	72.44±3.50	76.04±0.35	74.71±3.35	87.42±2.38
	20	80.04±2.54	72.02±2.82	74.55±3.09	79.98±0.96	91.33±0.36	91.29±0.67
GraphSAGE	1	43.04±14.01	48.81±11.45	55.53±12.71	27.65±2.39	61.35±1.35	45.36±7.13
	2	53.96±12.18	54.39±11.37	58.97±12.65	42.63±4.29	76.51±1.31	51.93±4.21
	5	68.14±6.95	64.79±5.16	66.07±6.16	64.83±1.62	89.06±0.69	78.26±1.93
	10	75.04±5.03	68.90±5.08	70.74±3.11	74.66±1.29	89.68±0.39	84.38±1.75
	20	82.07±2.03	71.52±4.11	76.49±1.75	73.66±2.87	90.31±0.41	88.61±1.18

cable to wide range of disciplines: biology, neuron science, engineering, and even in data processing. For example, brain networks (Varela et al., 2001), laser arrays (Kozyreff et al., 2000), generating music (Huepe et al., 2014), power-grid (Motter et al., 2013), wireless sensor networks (Tanaka et al., 2009)), consensus problems (Olfati-Saber et al., 2007), and data clustering as a method of unsupervised machine learning (Miyano and Tsutsui, 2007, 2008). For further references, we refer to Strogatz (2000); Acebrón et al. (2005); Dörfler and Bullo (2014) for excellent reviews.

6 CONCLUSION

We propose a new class of continuous-depth graph neural networks based on the classic Kuramoto model called Kuramoto Graph Neural Network (KuramotoGNN). We theoretically connect oversmoothing problem to the phase synchronization in a network of coupled oscillators, and we also show that with KuramotoGNN, the oversmoothing problem can be easily avoided. Through the empirical experiments, we showed that KuramotoGNN can have advantage in performance when compared to other variants of GNNs when the number of labeled data is limited and even when the input data is perturbed by Gaussian noise. It is interesting to note that this is just the classic version of the Kuramoto model, besides it, there are other variants, such as the time-delayed Kuramoto model, adaptive coupling strength functions, or second-order Kuramoto with damping term (Dörfler and Bullo, 2014). In future work, we intend to consider those variants of the Kuramoto model to GNN architectures.

7 ACKNOWLEDGEMENT

This work was supported by Toyo University Top Priority Research Program.

References

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017.

Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. Graph neural networks for social

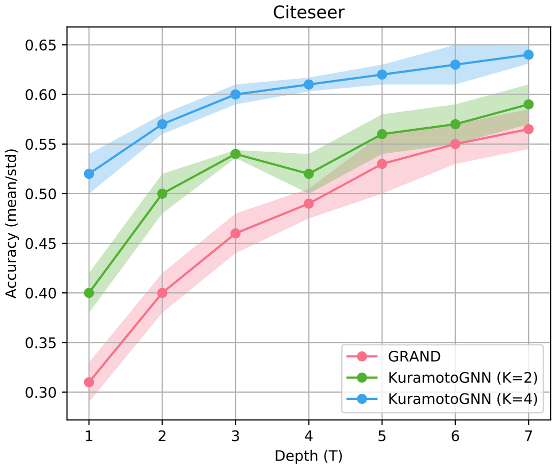

Figure 3: Change in performance at different depth on noisy Citeseer dataset.

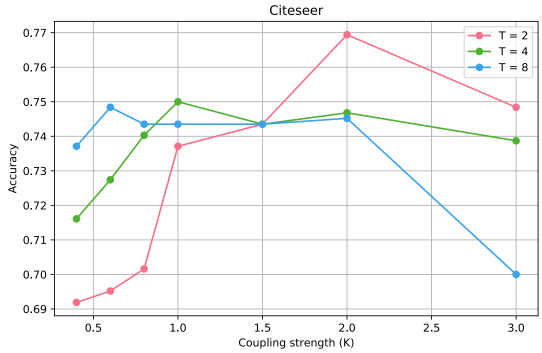

Figure 4: Change in performance of KuramotoGNN at different coupling strength (K) on Citeseer dataset.

recommendation. In *The world wide web conference*, pages 417–426, 2019.

Zhaoping Xiong, Dingyan Wang, Xiaohong Liu, Feisheng Zhong, Xiaozhe Wan, Xutong Li, Zhaojun Li, Xiaomin Luo, Kaixian Chen, Hualiang Jiang, et al. Pushing the boundaries of molecular representation for drug discovery with the graph attention mechanism. *Journal of medicinal chemistry*, 63(16):8749–8760, 2019.

Kenta Oono and Taiji Suzuki. Graph neural networks exponentially lose expressive power for node classification. *arXiv preprint arXiv:1905.10947*, 2019.

Hoang Nt and Takanori Maehara. Revisiting graph neural networks: All we have is low-pass filters. *arXiv preprint arXiv:1905.09550*, 2019.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Ben Chamberlain, James Rowbottom, Maria I Gorinova, Michael Bronstein, Stefan Webb, and Emanuele Rossi. Grand: Graph neural diffusion. In *International Conference on Machine Learning*, pages 1407–1418. PMLR, 2021.

Matthew Thorpe, Tan Minh Nguyen, Hedi Xia, Thomas Strohmer, Andrea Bertozzi, Stanley Osher, and Bao Wang. Grand++: Graph neural diffusion with a source term. In *International Conference on Learning Representations*, 2021.

T Konstantin Rusch, Ben Chamberlain, James Rowbottom, Siddhartha Mishra, and Michael Bronstein. Graph-coupled oscillator networks. In *ICML*, pages 18888–18909. PMLR, 2022.

Louis-Pascal Xhonneux, Meng Qu, and Jian Tang. Continuous graph neural networks. In *International Conference on Machine Learning*, pages 10432–10441. PMLR, 2020.

Yoshiki Kuramoto. Self-entrainment of a population of coupled non-linear oscillators. In Huzihiro Araki, editor, *International Symposium on Mathematical Problems in Theoretical Physics*, pages 420–422, Berlin, Heidelberg, 1975. Springer Berlin Heidelberg. ISBN 978-3-540-37509-8.

Francisco Varela, Jean-Philippe Lachaux, Eugenio Rodriguez, and Jacques Martinerie. The brainweb: phase synchronization and large-scale integration. *Nature reviews neuroscience*, 2(4):229–239, 2001.

Juan A. Acebrón, L. L. Bonilla, Conrad J. Pérez Vicente, Félix Ritort, and Renato Spigler. The kuramoto model: A simple paradigm for synchronization phenomena. *Rev. Mod. Phys.*, 77:137–185, Apr 2005. doi: 10.1103/RevModPhys.77.137.

Steven H Strogatz and Renato E Mirollo. Splay states in globally coupled josephson arrays: Analytical prediction of floquet multipliers. *Physical Review E*, 47(1): 220, 1993.

Daihai He and Lewi Stone. Spatio-temporal synchronization of recurrent epidemics. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 270 (1523):1519–1526, 2003.

Steven H. Strogatz. From kuramoto to crawford: exploring the onset of synchronization in populations of coupled oscillators. *Physica D: Nonlinear Phenomena*, 143(1): 1–20, 2000. ISSN 0167-2789. doi: https://doi.org/10.1016/S0167-2789(00)00094-4.

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

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *stat*, 1050:20, 2017.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Arthur T. Winfree. Biological rhythms and the behavior of populations of coupled oscillators. *Journal of Theoretical Biology*, 16(1):15–42, 1967. ISSN 0022-5193. doi: https://doi.org/10.1016/0022-5193(67)90051-3.

Ali El Ati and Elena Panteley. On frequency synchronization of kuramoto model with non-symmetric interconnection structure. In *CCCA12*, pages 1–6. IEEE, 2012.

Hassan K Khalil. Nonlinear systems third edition. *Patience Hall*, 115, 2002.

Ali Jadbabaie, Nader Motee, and Mauricio Barahona. On the stability of the kuramoto model of coupled nonlinear oscillators. In *Proceedings of the 2004 American Control Conference*, volume 5, pages 4296–4301. IEEE, 2004.

Nikhil Chopra and Mark W Spong. On exponential synchronization of kuramoto oscillators. *IEEE transactions on Automatic Control*, 54(2):353–357, 2009.

Andrew Kachites McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3(2):127–163, 2000.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3): 93–93, 2008a.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3): 93–93, 2008b.

Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. *arXiv preprint arXiv:1811.05868*, 2018.

Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 43–52, 2015.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

Eldad Haber and Lars Ruthotto. Stable architectures for deep neural networks. *Inverse problems*, 34(1):014004, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Liev Semiónovich Pontryagin, VG Boltyanskii, RV Gamkrelidze, EF Mishchenko, KN Trirogoff, and LW Neustadt. *LS Pontryagin Selected Works: The Mathematical Theory of Optimal Processes*. Routledge, 2018.

Patrick Kidger, James Morrill, James Foster, and Terry Lyons. Neural controlled differential equations for irregular time series. *Advances in Neural Information Processing Systems*, 33:6696–6707, 2020.

Xuanqing Liu, Tesi Xiao, Si Si, Qin Cao, Sanjiv Kumar, and Cho-Jui Hsieh. Neural sde: Stabilizing neural ode networks with stochastic noise. *arXiv preprint arXiv:1906.02355*, 2019.

Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. Augmented neural odes. *Advances in Neural Information Processing Systems*, 32, 2019.

Florian Dörfler and Francesco Bullo. Synchronization in complex networks of phase oscillators: A survey. *Automatica*, 50(6):1539–1564, 2014.

Gregory Kozyreff, AG Vladimirov, and Paul Mandel. Global coupling with time delay in an array of semiconductor lasers. *Physical Review Letters*, 85(18):3809, 2000.

Cristian Huepe, Marco Colasso, and Rodrigo F Cádiz. Generating music from flocking dynamics. In *Controls and Art*, pages 155–179. Springer, 2014.

Adilson E Motter, Seth A Myers, Marian Anghel, and Takashi Nishikawa. Spontaneous synchrony in power-grid networks. *Nature Physics*, 9(3):191–197, 2013.

Hisa-Aki Tanaka, Hiroya Nakao, and Kenta Shinohara. Self-organizing timing allocation mechanism in distributed wireless sensor networks. *IEICE Electronics Express*, 6(22):1562–1568, 2009.

Reza Olfati-Saber, J Alex Fax, and Richard M Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.

Takaya Miyano and Takako Tsutsui. Data synchronization in a network of coupled phase oscillators. *Physical review letters*, 98(2):024102, 2007.

Takaya Miyano and Takako Tsutsui. Collective synchronization as a method of learning and generalization from sparse data. *Physical Review E*, 77(2):026112, 2008.

Supplementary Materials

1 EXPERIMENTAL DETAILS AND MORE RESULTS

For all six graph node classification datasets, including CORA, CiteSeer, PubMed, coauthor graph CoauthorCS, and Amazon co-purchasing graphs Computer and Photo, we consider the largest connected component. Table 1 lists the fine-tuned T, and Table 2 lists the fine-tuned coupling-strength K for the results in the main paper.

Although the depth values in KuramotoGNN is generally smaller than GRAND-l, our model is more complex and require more steps (using adaptive ODE solvers) to calculate the solution. As the result, the number of actual layers (number of discretisatized steps) of our model is equal or larger than GRAND-l.

Table 1: fine-tuned depth T for KuramotoGNN and GRAND-l.

Model	CORA	Citeseer	Pubmed	CoauthorCS	Computer	Photo
KuramotoGNN	12	5	8	0.8	1	1.5
GRAND-l	18.2948	7.8741	12.9423	3.2490	3.5824	3.6760

Table 2: fine-tuned coupling strength K for KuramotoGNN.

Dataset	Coupling strength K
CORA	1
Citeseer	2
Pubmed	0.9
CoauthorCS	1.8
Computer	4
Photo	2

We also further explore the effects of the depth and coupling strength for KuramotoGNN by conducting further experiements based on various depths and coupling strengths. Table 3 shows the performances in accuracy of KuramotoGNN on three datasets: CORA, Citeseer, and Pubmed. Overall, the coupling strength is more sensitive in case of small depths, but in larger depths, the chances in performances are not significant between choices of coupling strengths.

Table 3: Mean and std of classification accuracy of KuramotoGNN in different depths and coupling strengths on three CORA, Citeseer, and Pubmed graph node classification tasks. (Unit: %)

Depth T	Coupling Strength K	CORA	Citeseer	Pubmed
2	0.7	75.13±1.35	70.24±3.41	76.81±1.69
	0.8	76.27±2.89	71.85±2.16	78.07±1.77
	0.9	79.8±0.77	73.87±1.63	79.85±1.19
	1	78.15±0.98	70.89±1.81	77.61±2.22
3	0.7	75.89±1.77	72.42±2.26	76.88±2.58
	0.8	78.43±1.08	76.33±2.48	79.34±1.48
	0.9	79.67±0.77	72.58±2.91	78.77±0.99
	1	79.54±2.14	74.8±1.19	79.13±0.99
5	0.7	82.03±1.79	72.54±1.31	77.98±2.95
	0.8	79.37±0.49	72.58±2.77	79.46±0.48
	0.9	81.98±1.96	74.88±1.22	78.91±2.47
	1	82.92±0.88	73.99±0.84	80.46±1.82
8	0.7	81.37±1.13	74.56±1.65	80.17±0.80
	0.8	82.49±0.74	75.24±2.39	79.75±1.28
	0.9	82.77±1.29	75.4±2.43	80.07±0.57
	1	83.22±1.57	75.04±0.7	78.49±3.03
10	0.7	82.26±1.05	74.4±3.4	79.49±1.16
	0.8	82.49±0.98	75.93±1.18	79.27±0.52
	0.9	81.6±0.98	74.56±0.77	78.17±1.86
	1	83.43±1.3	75.12±1.02	79.08±1.93
12	0.7	83.53±0.72	74.88±1.87	78.84±1.69
	0.8	83.83±0.59	75.93±1.44	79.9±0.95
	0.9	81.75±1.61	74.35±1.99	79.93±0.52
	1	85.18±1.35	73.63±1.72	79.35±0.8
16	0.7	84.06±2.46	73.55±0.96	77.49±1.41
	0.8	82.06±2.05	74.68±1.17	78.67±1.36
	0.9	83.35±0.48	76.01±1.45	75.77±0.12
	1	82.82±1.13	75.16±1.19	75.51±2.18
18	0.7	84.37±0.68	75.16±0.94	78.46±2.46
	0.8	82.97±0.75	75.28±1.21	76.60±2.15
	0.9	82.99±0.44	73.83±1.95	75.67±1.63
	1	82.79±0.38	75.4±1.49	74.2±1.71