FDGATII: Fast Dynamic Graph Attention with Initial Residual and Identity

Gayan K. Kulatilleke(✉), Marius Portmann, Ryan Ko, and Shekhar S. Chandra

School of Information Technology and Electrical Engineering, University of Queensland, Queensland, Australia
{ryan.ko,shekhar.chandra}@uq.edu.au

Abstract. Despite their recent popularity, deep and efficient Graph Neural Networks remain a major challenge due to (a) over-smoothing, (b) noisy neighbours (heterophily), and (c) the suspended animation problem. Inspired by the attention mechanism's ability to focus on selective information, and prior work on feature preserving mechanisms, we propose FDGATII, a dynamic deep-capable model that addresses all these challenges *simultaneously* and efficiently. Specifically, by combining Initial Residuals and Identity with the more expressive dynamic self-attention, FDGATII effectively handles noise in heterophilic graphs and is capable of depths over 32 with no over-smoothing, overcoming two main limitations of many prior GNN techniques. By using edge-lists, FDGTII avoids computationally intensive matrix operations, is parallelizable and does not require knowing the graph structure upfront. Experiments on 7 standard datasets show that FDGATII outperforms the GAT and GCN based benchmarks in accuracy and performance on fully supervised tasks. We obtain State-of-the-art (SOTA) on the highly heterophilic Chameleon and Cornell datasets with 1 layer, and come only 0.1% short of Cora SOTA with zero graph pre processing. https://github.com/gayanku/FDGATII

Keywords: Dynamic attention · Heterophily · Over-smoothing

1 Introduction

Recently, research on graphs has been receiving increased attention due to the great expressive power and pervasiveness of graph structured data [29]. Many interesting irregular domain tasks such as 3D meshes, social networks, telecommunication networks and biological networks involve data that are not representable in grid-like structures [25]. As a unique non-Euclidean data structure for machine learning, graphs can be used to represent diverse feature rich domains.

A Graph Neural Network (GNN) generalizes deep neural networks (DNNs) from regular structures to irregular graph data. GNNs perform neighbourhood structure aggregation and node feature transformation to map nodes to low dimensional embeddings [15,17], mostly differing in how aggregation and combination

H. Aziz et al. (Eds.): AI 2022, LNAI 13728, pp. 73–86, 2022.
https://doi.org/10.1007/978-3-031-22695-3_6

is performed [4]: Graph Convolutional Network (GCN) [13] uses convolution [16]; Graph Attention Network (GAT) [25] uses attention; GraphSage [8] uses max pooling. Downstream tasks such as node classification, clustering, and link prediction [8,22] use these aggregated low dimensional vectors [28].

Most graphs require the interaction between nodes that are not directly connected, i.e., higher-order information which is achieved by stacking GNN layers [2]. However, stacking layers degrades the performance [5,20] due to over-smoothing: node representations become indistinguishable with increasing number of layers [6,13,26]. Further, GNNs in general are not able to handle long-range information due to over-squashing: information from the exponentially growing receptive field being compressed into fixed-length node vectors [2] due to its unfocused aggregation mechanism. Finally, deeper models stop responding to training due to the suspended animation problem [26], i.e. depth is a problem [6].

To avoid these problems, several works combine deep propagation with shallow neural networks; SGC [26] used the K-th power of the adjacency matrix to capture higher-order information; H2GCN [29] aggregates higher-order information at each round. However, this form of linear combination of neighbour features at each layer looses the powerful expression ability of deep nonlinear architectures, essentially making them shallow models [5].

In another attempt to address the problem and incorporate deeper layers, JKNet [27] used dense skip connections, DropEdge [23] randomly removed graph edges and GCNII [5] added a portion of Initial residual and Identity. GCNII showed remarkable results for up to 64 layers and is the SOTA (Table 2) in Cora, a homophilic benchmark dataset. However, all these are spectral approaches based on the Laplacian eigenbasis and requires the whole graph structure [25]. The normalization used is computationally expensive and not scalable.

Furthermore, due to naive uniform aggregation of the neighbourhood, most of these models, including GCNII, are more suitable for homophilic datasets, where nodes linked to each other are more likely to belong in the same class, i.e., neighbourhoods with low noise. In practice, real-world graphs are also often noisy with connections between unrelated nodes [12], resulting in poor performance in current GNNs. As many popular GNN models implicitly assume homophily, results may be biased, unfair or erroneous [19]. This can result in a 'filter bubble' phenomenon in a recommendation system (reinforcing existing beliefs/views, and downplaying the opposite ones), or making minority groups less visible in social networks [29]. As a result, despite GCNIIs SOTA in homophilic datasets (Cora), its accuracy in heterophilic datasets (Texas, Wisconsin) is relatively poor [29].

On the other hand, [24] showed that self-attention is sufficient for achieving SOTA performance. GAT [25] generalizes attention for graphs using attention-based neighbourhood aggregation. Importantly, GAT improves on simple averaging [13] and max pooling [8] by allowing every node to compute a weighted average of its neighbours [4], which is a form of selective aggregation. The generalization ability of the attention mechanism helps GNNs generalize to larger and more noisy graphs [14]. By determining individual attention on each neighbour, GAT ignores irrelevant neighbours and focuses on those that are relevant [2].

Surprisingly, yet, GATs heterophilic performance is poor (Table 2).

A refinement, GATv2 [4], uses a more expressive dynamic attention, where the ranking of attended nodes is better conditioned on the query node by replacing the supposedly monotonic GAT attention function with a universal approximator attention function that is strictly more expressive. However, GAT or GATv2 *alone, in its current form* cannot handle heterophilic data due to the still present essentially local aggregation operation [17].

In Table 2, under heterophily, only H2GCN outperforms a Multilayer Perceptron (MLP) of 1 layer which uses only node features and no structural information. Furthermore, most GNN models use simple graph convolution based aggregation schemes [8,13], leading to filter incompleteness. While this can be solved by using a more complex graph kernel [1], currently, even attention-based models perform poorly given heterophilic data, despite the ability to focus on the most "relevant" content.

Thus, it remains an open problem to design efficient GNN models that effectively handle (a) over-smoothing, (b) suspended animation and (c) heterophily/noise simultaneously. As observed by [5], it is even unclear whether the network depth is a resource or a burden when designing new GNNs. Motivated by these limitations, we propose a generalizable, efficient, and parallelizable attention based deep-capable model that addresses aforementioned challenges simultaneously. Our main contributions are:

- We introduce a novel deep-capable GNN model, FDGATII, successfully combining strengths of GCN and GAT worlds by using dynamic attention supplemented with Initial residual and Identity, capable of handling the major graph challenges: over-smoothing, noisy neighbours (heterophily) and suspended animation *simultaneously*. To the best of our knowledge, this is the first time a graph attentional model has demonstrated depths of up to 32, a limitation of many prior GNN techniques, attention based or otherwise, and show that dynamic attention is better suited for heterophilic datasets, if used with modifications.
- FDGATII is computationally efficient. It does not require an adjacency matrix as input nor its subsequent, expensive matrix operations or normalizations. Further, its attention layers can be parallelized across edges while feature computation can be parallelized across all nodes.
- FDGATII has the same complexity as SOTA GCN models, but uses significantly fewer layers to achieve comparable or better results, yielding a superior efficiency-to-accuracy ratio across homophilic and heterophilic datasets.

Extensive experiments on 7 benchmarks show that FDGATII outperforms GAT and GCN based benchmarks in accuracy as well as on accuracy vs efficiency, on fully supervised tasks. FDGATII achieves SOTA accuracy results on Chameleon and Cornell datasets, beating H2GCN, a model specifically designed for heterophily. There is zero graph pre processing. FDGATII consumes over a magnitude less computational resources and is only –0.1% below SOTA for Cora, placing a close second. By not assuming homophily, FDGATII minimises its potential negative effects: bias, unfairness and potential for filter bubbles. FDGATII is also capable of inductive learning. Table 1 has a full feature comparison.

Table 1. Feature comparison: GAT, GCN, GCNII and FDGATII. *Cham & Cornell

Feature	GAT	GCN	GCNII	FDGATII
No graph pre processing (ex: normalisation)	Yes	No	No	Yes
Does not require knowing graph structure upfront	Yes	No	No	Yes
Processing can be parallelized	Yes	No	No	Yes
Free from oversmoothing	No	No	Yes	Yes
Free from suspended animation	No	No	Yes	Yes
Heterophilic performance	Poor	Poor	Good	SOTA*
Capable of deep architectures (layers > 8)	No	No	Yes	Yes
Dynamic attention	No	No	No	Yes
Inductive learning	Yes	No	Yes	Yes
# layers for best Cora accuracy	2	2	64	2

2 Related Work

2.1 Notation

$G = (V, E)$ is an undirected graph with n nodes $v_j \in V$ and m edges $(v_i, v_j) \in E$. $\bar{G} = (V, \bar{E})$ is its self-looped graph. A is the adjacency matrix, D the degree matrix of G. Adjacency matrix and degree matrix of $\bar{G}$ is $\bar{A} = A + I$ and $\bar{D} = D + I$. The symmetric positive semi definite *normalized graph Laplacian matrix* is given by $L = I_n - D^{-1/2}AD^{-1/2}$ with eigen-decomposition $U\Lambda U^T$. Λ is its diagonal eigenvalue matrix, $U \in R^{n \times n}$ is the unitary eigenvector matrix.

2.2 Convolution and GCN

Given signal x and filter $g_\gamma(\Lambda) = diag(\gamma)$ the graph convolution operation is $g_\gamma(L) * x = U g_\gamma(\Lambda) U^T x$ where $\gamma \in R^n$ is the vector of spectral filter coefficients. $g_\gamma(\Lambda)$ can be approximated by a truncated expansion of a K^{th} order Chebyshev polynomial [9], where $\theta \in \mathbf{R}^{K+1}$ corresponds to a vector of polynomial coefficients:

$$\mathbf{U} g_\theta(\Lambda) \mathbf{U}^T \mathbf{x} \approx \mathbf{U} \left(\sum_{l=0}^{K} \theta_l \mathbf{\Lambda}^l \right) \mathbf{U}^T \mathbf{x} = \left(\sum_{l=0}^{K} \theta_l \mathbf{L}^l \right) \mathbf{x} \tag{1}$$

GCN [13] simplifies graph convolution by *fixing* $K = 1, \theta_0 = 2\theta$ and $\theta_1 = -\theta$ to get $g_\theta * x = \theta(I + D^{-1/2}AD^{-1/2})x$ and uses a normalized adjacency matrix, $\bar{P} = \bar{D}^{-1/2}\bar{A}\bar{D}^{-1/2} = (D+I_n)^{-1/2}(A+I_n)(D+I_n)^{-1/2}$. Each GCN layer (Eq. 2) contains a nonlinear activation function σ, typically ReLU.

$$\mathbf{H}^{l+1} = \sigma\left(\bar{\mathbf{P}}\mathbf{H}^l\mathbf{W}^l\right) \tag{2}$$

However, node embeddings are aggregated recursively layer by layer. Embeddings in the final layer requires all previous embeddings, resulting in high memory cost. GCN gradient update in the full-batch training scheme needs storing all intermediate embeddings, limiting scalability. As the learned filters depend on the Laplacian eigenbasis, which depends on the entire graph structure, a model trained on a graph cannot be directly applied to a different graph structure [25].

2.3 GCNII

GCNII [5] extends the fixed coefficient GCN to a deep model by expressing the K order polynomial filter as *arbitrary* coefficients using Initial residual and Identity (II). Essentially, GCNII 1) combines the preprocessed (normalized) representation $\bar{\mathbf{P}}\mathbf{H}^l$ with an initial residual connection from the first layer $\mathbf{H}^0$; and 2) adds an identity $\mathbf{I}_n$ to the l-th weight matrix $\mathbf{W}^l$. By using a connection to the initial residual $\mathbf{H}^0$, GCNII ensures that the final representation of each node retains at least a α_l fraction from the input layer.

However, as GCNII combines neighbour embeddings by uniformly averaging, its heterophilic performance is relatively poor. GCNs preserve structure over features, regardless of the graph's heterophilic nature, resulting in original node features being destroyed [11]. Further, [20] showed that GCNs tend to fail when graphs are dense and do not always improve with more layers. Alternatively, a selective aggregation of the neighbourhood allows focusing on relevant nodes [29].

2.4 Attention Mechanism and GAT

The DP (dot-product) attention mechanism (Equation 3) [18,24] has been widely used in GNNs [12,28]. Different from DP, GAT [25] uses concatenation followed by a 1-layer feed-forward network parameterized by $\mathbf{a}$ (Eq. 4).

$$e\left(\mathbf{h}_i, \mathbf{h}_j\right) = \text{LeakyReLU}((\mathbf{W}\mathbf{h}_i)^T \cdot \mathbf{W}\mathbf{h}_j) \tag{3}$$

$$e\left(\mathbf{h}_i, \mathbf{h}_j\right) = \text{LeakyReLU}\left(\mathbf{a}^T \cdot [\mathbf{W}\mathbf{h}_i \parallel \mathbf{W}\mathbf{h}_j]\right) \tag{4}$$

In contrast to GCN, which weighs all neighbours $j \in \mathcal{N}_i$ with equal importance, GAT computes a learned weighted average of the representations of $\mathcal{N}_i$ using attention. Compared to GCN, assigning different weights for neighbours can mitigate noise and achieve better results [28] while being more robust in the presence of noisy "irrelevant" neighbours [2].

3 Proposed Architecture

Our proposed design (Fig. 1) is built upon a local embedding step that extracts local node embeddings from feature vectors using GATv2. To extend GATv2 to handle heterophilic and noisy data, we borrow two techniques from GCNII [5] and H2GCN [29] with modifications, namely residual connection and identity.

However, the theoretical foundation of our model, which is grounded in the spatial domain, is completely different from GCNII which is spectral. We do not require edge values; only the presence or absence of an edge: i.e. a simple list of edges. Using only the edge-list as [25], with self-loops as [10,13], we avoid computationally intensive matrix operations such as inversions or eigen-decompositions and the need to know the graph structure upfront. Experiments show our design is efficient, robust and generalizes well to homophilic and heterophilic datasets alike.

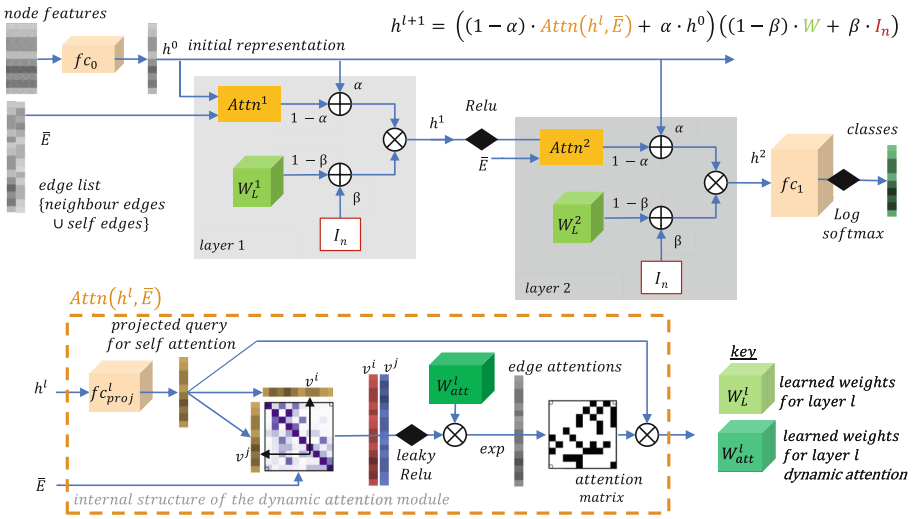

Fig. 1. FDGATII uses dynamic attention to combine relevant neighbours via edge-lists, an $\alpha\%$ of initial representation h^0 projected via fc_0 and a $\beta\%$ of Identity I_n at each layer. Attention module concatenates source (row) and destination (column) features of each edge, projects via W_H^n, applies a non-linearity (leaky-relu) and an $exp()$ to obtain the edgewise attentions before reshaping to a matrix suitable for softmax with the query. After multiple layers, an fc_1 projection and log softmax provides the node classification.

Typically, GNN models follow an iterative learning approach:

$$\mathbf{h}_i^{1+1} = \text{COMBINE}\Big(\mathbf{h}_i^l, \text{AGG}(\{\mathbf{h}_j^l : j \in N_i\})\Big),$$
$$\mathbf{h}_i^0 = \mathbf{X}_i, \text{and } y_i = \arg\max\{\text{softmax}(\mathbf{h}_i^K)\mathbf{W}\}$$

where, AGG is a permutation invariant aggregation operator and COMBINE is a learnable function. By adding self-nodes, we amalgamate COMBINE and AGG to simplify the process and apply a more expressive attention operator ATTN to both tasks simultaneously, defined by:

$$\mathbf{h}_i^{1+1} = \text{ATTN}(\{\mathbf{h}_j^l : j \in N_i \cup i\})$$

3.1 Initial Residual and Identity (II)

We incorporate initial representation $\mathbf{H}^0$ and identity $\mathbf{I}_n$, in α_l and β_l fractions, with edge-list $\bar{\mathbf{E}}$ to formally define the $(l+1)$-th layer of FDGATII as:

$$\mathbf{H}^{l+1} = \sigma\left[\Big((1-\alpha_l)\,\text{ATTN}(\bar{\mathbf{E}}, \mathbf{H}^l) + \alpha_l \mathbf{H}^0\Big)\cdot\Big((1-\beta_l)\,\mathbf{I}_n + \beta_l \mathbf{W}^l\Big)\right] \tag{5}$$

According to [10], identity mapping of the form $\mathbf{H}^{l+1} = \mathbf{H}^l(\mathbf{W}^l + \mathbf{I}_n)$, as in Eq. 5, satisfies the following properties: 1) the optimal weight matrices $\mathbf{W}^l$ have

small norms; 2) the only critical point is the global minimum. The first property allows us to put strong regularization on $\mathbf{W}^l$ to avoid over-fitting, while the latter is desirable in semi-supervised tasks where training data is limited.

Next, it is theoretically proven [20] that a K-layer GNN's convergence rate depends on s^K, where s is the maximum singular value of the weight matrices $\mathbf{W}^l, l = 0, \ldots, K-1$. By replacing $\mathbf{W}^l$ with $(1-\beta_l)\mathbf{I}_n + \beta_l \mathbf{W}^l$ and regularizing $\mathbf{W}^l$, resulting singular values of $(1-\beta_l)\mathbf{I}_n + \beta_l \mathbf{W}^l$ stay closer to 1, which implies that s^K is large, and the information loss is relieved.

3.2 Selection of Proper Attention

It has been shown that GAT is better at learning label-agreement between a target node and its neighbors than DP attention [12]. Variance of GAT depends only on the norm of features, while the DP variance depends on the variance of the input's dot-product and the expectation of the square of the input's dot-product. As a result, with more layers, more features of i and j correlating resulting in a larger dot-product and the subsequent softmax normalization which increases the larger values further, DP is only able to attend to a small set of neighbours.

3.3 Dynamic Attention (GATv2)

According to [4], the main problem in the standard GAT scoring function (Eq. 4) is that the learned layers $\mathbf{W}$ and $\mathbf{a}$ are applied consecutively, and thus can be collapsed into a single linear layer. GATv2 replaces the linear approximator with a universal approximator (Eq. 6) and has been shown to perform better on noisy data [4]. Further, theoretically, DP is strictly weaker than GATv2. We use this form of dynamic attention for our aggregation function.

Specifically, a scoring function $e : R^d \times R^d \rightarrow R$ computes a score for every edge (j, i), which indicates the importance of the features of the neighbour j to the node i:

$$e\left(\mathbf{h}_i, \mathbf{h}_j\right) = \mathbf{a}^T \cdot \text{LeakyReLU}\left(\mathbf{W}\left[\mathbf{h}_i \parallel \mathbf{h}_j\right]\right), \tag{6}$$

where attention scores $\mathbf{a} \in R^{2d'}$ and weights $\mathbf{W} \in R^{d' \times d}$ are learned. $\parallel$ denotes vector concatenation. We capture the graph structure using edges, computing $e_{i,j}$ for *all* $j \in N_i$ neighbourhood of node i. Attention scores are normalized across all connected sparse neighbours $j \in \mathcal{N}_i$ using softmax.

$$\alpha_{ij} = softmax_j\left(e\left(\mathbf{h}_i, \mathbf{h}_j\right)\right) = \frac{exp\left(e\left(\mathbf{h}_i, \mathbf{h}_j\right)\right)}{\sum_{j' \in \mathcal{N}_i} exp\left(e\left(\mathbf{h}_i, \mathbf{h}_{j'}\right)\right)} \tag{7}$$

Finally, we compute the weighted average of the transformed features of the neighbour nodes (followed by a nonlinearity σ) as the new representation of i, using the normalized attention coefficients:

$$\text{ATTN}_i^{'} = \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij} \mathbf{W}\mathbf{h}_j\right) \tag{8}$$

In addition to Eq. 5, following [5], we also propose FDGATII* with dual weight matrices for smoother representation, defined as:

$$\begin{aligned}\mathbf{H}^{l+1} = \sigma\Big[(1-\alpha_l)\mathrm{ATTN}(\bar{\mathbf{E}},\mathbf{H}^l)\left((1-\beta_l)\mathbf{I}_n + \beta_l\mathbf{W}_1^l\right) \\ +\alpha_l\mathbf{H}^0\left((1-\beta_l)\mathbf{I}_n + \beta_l\mathbf{W}_2^l\right)\Big]\end{aligned} \tag{9}$$

GCNII [5] uses β_l is to ensure the decay of the weight matrix adaptively increases with more layers. While FDGATII typically achieves best accuracy early with a few layers, we still adopt the same mechanism, $\beta_l = log\left(\frac{\lambda}{l}+1\right) \approx \frac{\lambda}{l}$, where λ is a hyperparameter, for robustness at high depth. Following [27], we add skip connections in the form of initial representations H^0 as in [5].

FDGATII differs from existing models with respect to its use of a modified attention mechanism. Notably, we demonstrate competitive performance of GATv2+II with only a few layers in non-homophilous networks. Using edge-lists avoids computationally intensive matrix operations. Table 1 summarizes how FDGATII accumulates all benefits from GCN and GAT worlds with none of the drawbacks.

3.4 Datasets and Experiments

Homophily is the fraction of edges which connect two nodes of the same label [17]. A higher value (1) indicates strong homophily; a lower value (0) indicates strong heterophily.

We evaluate FDGATII against SOTA GNNs on benchmark graph datasets for fully supervised classification. Following [5,21], we use 7 datasets (Table 5). Cora, Citeseer and Pubmed are homophilic citation networks where nodes correspond to documents, and edges correspond to citations. The remaining four are heterophilic datasets of web networks, where nodes and edges represent web pages and hyperlinks, respectively. Node feature vectors are bag-of-word representations of the document. Following [5,21] we use the same data splits, 60:20:20 nodes for training:validation:testing, learning rate = 0.01, hidden units = 64 and measure the average performance on the 10 splits for each dataset.

We choose GCNII [5] as our performance and accuracy benchmark as it is (a) more current; (b) most similar to our work in the use of initial representation and identity; (c) actively attempts to solve over smoothing (d) is the SOTA in Cora (a prominent dataset for GNN model comparison) and *most* importantly (d) it is a deep-capable model. We also compare with H2GCN [29] which is the SOTA for Cornel, Texas and Wisconsin; highly heterophelic datasets, but note that H2GCN is a shallow model.

For training and inference time measurements we perform GPU warm-up and synchronization prior to measurements. We take the average time for 1000 inferences to lower any possibility of errors and to be more reflective of real-world use of models. We ignore pre processing times, but point out, unlike the benchmarks, FDGATII has *no* expensive full graph eigen operations or normalizations.

Table 2. Mean classification accuracy of full-supervised node classification. (a) reported by [5], (b) reported by [29], (c) best results running GCNII (official author implementation) and H2GCN (public pytorch repo: github.com/GitEventhandler/H2GCN-PyTorch) on data splits of [5], (d): our FDGATII, with same splits. Best is bold and second underlined. # of layers in parenthesis.

Dataset	Cora	Cite.	Pumb.	Cham.	Corn.	Texa.	Wisc.
Hormophily %	0.81	0.74	0.8	0.23	0.30	0.11	0.21
MLP[b]	74.75(1)	72.41(1)	86.65(1)	46.36(1)	81.08(1)	81.89(1)	85.29(1)
GCN[a]	85.77	73.68	88.13	28.18	52.70	52.16	45.88
GAT[a]	86.37	74.32	87.62	42.93	54.32	58.38	49.41
Geom-GCN-I[a]	85.19	**77.99**	90.05	60.31	56.76	57.58	58.24
GraphSAGE[b]	86.90	76.04	88.45	58.73	81.18	82.43	75.95
MixHop[b]	87.61	76.26	85.31	60.50	75.88	77.84	75.88
H2GCN-1[b]	86.92	77.07	89.40	57.11	82.16	**84.86**	**86.67**
APPNP[a]	87.87	76.53	89.40	54.3	73.51	65.41	69.02
JKNet[a]	85.25(16)	75.85(8)	88.94(64)	60.07(32)	57.30(4)	56.49(32)	48.82(8)
JKNet(Drop)[a]	87.46(16)	75.96(8)	89.45(64)	62.08(32)	61.08(4)	57.30(32)	50.59(8)
Incep(Drop)[a]	86.86(8)	76.83(8)	89.18(4)	61.71(8)	61.62(16)	57.84(8)	50.20(8)
GCNII[a]	**88.49(64)**	77.08(64)	89.57(64)	60.61(8)	74.86(16)	69.46(32)	74.12(16)
GCNII*[a]	88.01(64)	77.13(64)	90.30(64)	62.48(8)	76.49(16)	77.84(32)	81.57(16)
H2GCN-1[c]	77.3038	74.5220	87.5887	49.0351	73.7838	78.9189	79.0196
GCNII[c]	88.2696	76.9325	90.3499	63.7500	77.2973	78.3784	79.8039
FDGATII[d]	88.3903(2)	76.3082(1)	**90.5502(2)**	**66.1184**(1)	**84.3243**(1)	83.7838(1)	86.0784(1)

Table 3. Inductive learning - F1 (micro) on PPI. (1): Results from [5]. (2): Our results with identical settings and Eq. 5. Note, we do not require any data pre processing

Method	PPI(reported)[1]	Method	PPI(our tests)[2]
GraphSAGE	61.2	FDGATII (2 layers)	98.51
GAT	97.3	FDGATII (3 layers)	98.91
JKNet	97.6	FDGATII (4 layers)	99.18
GeniePath	98.5	FDGATII (5 layers)	99.17
Cluster-GCN	99.36	FDGATII (6 layers)	99.24
GCNII (9 layers)	99.53	GCNII (9 layers)	99.52
GCNII* (9 layers)	**99.56**	GCNII* (9 layers)	99.53

4 Results and Discussion

4.1 Fully Supervised Node Classification

Table 2 reports the mean classification accuracy. We reuse the metrics already reported by [5] and [29]. We observe that FDGATII demonstrates SOTA results on heterophilic datasets while still being competitive on the homophilic datasets. Further, FDGATII exhibits significant accuracy increases over its attention based predecessor, GAT. This result suggests that dynamic attention with initial residuals and identity improves the predictive power whilst keeping the layer count (and hence the model parameters and computational requirements) low.

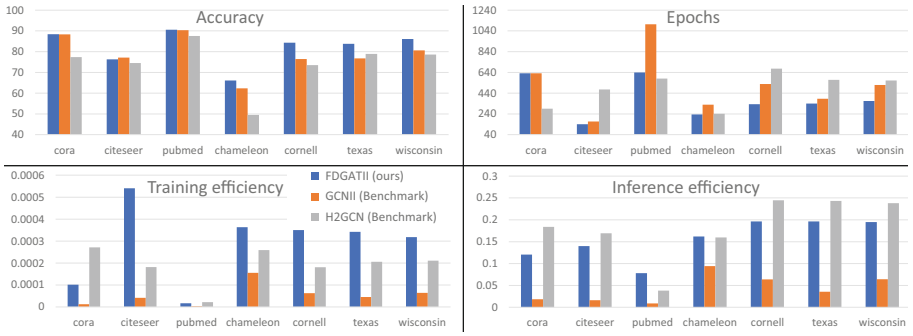

Fig. 2. Accuracy, epochs, training and inference time comparison. For variants, we use the lowest average time taken to run all 10 standard splits. Efficiency = 1/time. Original GCNII is in pytorch. Original H2GCN is in TF. A public pytorch H2GCN is used to eliminate any framework effects. Tested on Google colab with GPU.

Table 4. Ablation study w/0 II and w/0 dynamic attention. * Eq. 5, ** Eq. 9. Hyperparameter settings from [5]. $L1$ and $L2$ are 1 and 2 layers, respectively.

Metric	Cora	Cite.	Pumb.	Cham.	Corn.	Texa.	Wisc.
Without II, L1	86.90	**75.65**	87.01	**65.18**	65.95	62.16	54.51
Without II, L2	86.74	74.45	86.19	49.78	58.92	57.30	51.76
With II*, L1	87.06	75.07	89.96	61.34	76.76	70.00	81.96
With II*, L2	**87.79**	74.88	**90.35**	47.82	79.19	79.73	83.53
With II*, L2 w/o dynamic attention	87.53	75.01	90.34	46.00	79.73	81.90	82.54
With II**, L1	84.91	75.28	89.48	49.12	80.27	78.65	84.12
With II**, L2	86.52	75.14	90.12	44.34	**80.81**	**82.16**	84.90
With II**, L2 w/o dynamic attention	85.98	74.48	90.03	43.99	80.80	80.54	**85.49**

4.2 Inductive Learning

We use the PPI dataset and follow [8] using 20:2:2 graphs for train:validation:test. For settings, we follow [5]: 2048 hidden units, learning rate 0.001. Similar to [5,25], we add a skip connection from layer l to $l+1$. Table 3 reports the F1 (micro) scores. Results show that FDGATII is capable of competitive inductive learning.

4.3 Ablation Study

In this section, along with Table 4, we consider the effect of various design strategies. Our 1 or 2-layer models, without Initial residual and Identity (II), is theoretically equivalent to GAT(static attention)/GATv2(dynamic attention). The ablation study indicates that the addition of II *together with dynamic attention* results in improvements on the heterophilic dataset performance. This result suggests that both II and dynamic attention techniques are needed to solve the problem of over-smoothing and data heterophily. Figure 4 also confirms GAT/GATv2 cannot handle heterophily or depth unaided, while FDGATII shows significant and consistently better results.

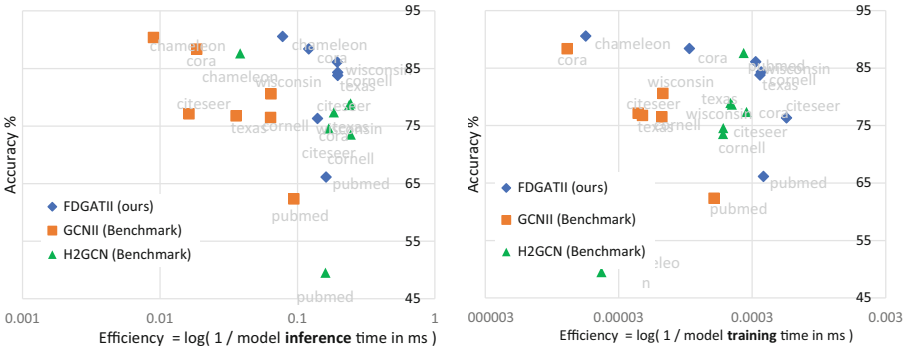

Fig. 3. Efficiency vs accuracy, on GPU with warm-up. **left:** average inference time for 1000 iterations. **right:** average training efficiency for 10 iterations. Efficiency = log(1/time). Top-right is better.

4.4 Performance and Efficiency

Figure 3 summarizes the high accuracy-to-computational-time-efficiency ratio of FDGATII clearly indicating its superior performance mix. The proposed architecture performs consistently better across noisy and diverse datasets with comparable or better accuracy (Table 2) while exhibiting superiority in training and inference times, specifically 12x faster training speeds and up to 9x faster inference speeds over our chosen deep-capable SOTA benchmark, GCNII [5]. FDGATII is 3x faster than H2GCN [29] on Citeseer. Our dynamic attention achieves higher expressive power with fewer layers paying selective attention to nodes, while the II supplements self-node features in highly heterophilic datasets.

By using edge-lists, FDGATII avoids computationally intensive eigen decompositions and matrix operations as well as the need to know the graph structure upfront. Also, output feature computation can be parallelized across nodes while the attention computation can be parallelized across all edges. While FDGATII has the same time complexity of GCNII, by using significantly fewer layers (Table 2 and Table 5), it achieves comparable or better results with superior efficiency-to-accuracy ratios. Note, in Fig. 2, the graph pre processing (inversion, normalization) times for benchmarks were not taken into account due to focus on model training and inference. FDGATII has zero graph pre processing.

4.5 Suspended Animation and Over Smoothing

Responding to training indicates absence of suspended animation [26], while effectively handling higher receptive fields indicates robustness to over-smoothing [6]. Figure 4 shows FDGATII's performance for 3 selected datasets under increasing layer depth. There is no evidence of performance degradation from suspended animation or over smoothing even at depth of 32. Accuracy is achieved early and sustained over higher depths. In Cora, the drop is 0.1 for 32 layers. H2GCN reported OOM for depths over 8.

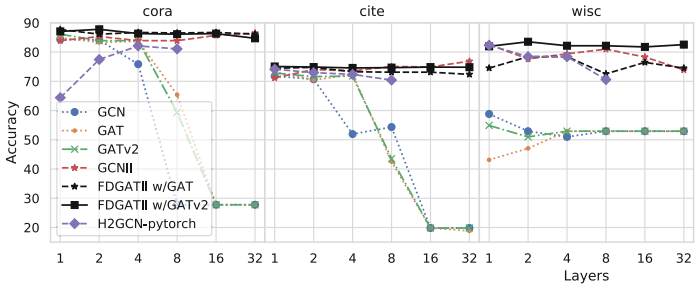

Fig. 4. Accuracy vs layer depth (on Goole Colab with GPU). FDGATII is consistent. H2GCN OOM after 8 layers. Depth and heterophily degrades GAT/GATv2 accuracy.

4.6 Broader Issues Related to Heterophily

Many popular GNN models implicitly assume homophily, producing results that may be biased, unfair or erroneous [29]. This can result in the so-called 'filter bubble' phenomenon in a recommendation system (reinforcing existing beliefs/views, and downplaying the opposite ones), or make minority groups less visible in social networks, creating ethical implications [7]. FDGATII's novel self-attention mechanism, where dynamic attention supplemented with II for feature preservation, reduces the filter bubble phenomenon and its potential negative consequences, ensuring fairness and less bias.

This offers new possibilities for future research into data where 'opposites attract', in which the majority of linked nodes are different, such as social and dating networks (the majority of persons of one gender connect with the opposite gender), chemistry and biology (amino acids bond with dissimilar types in protein structures), e-commerce (sellers with promoters and influencers), and dark web and other cybercrime related activities [29]. In a typical dark web social network, fraudsters are more likely to connect to intermediaries and prospective victims than to other fraudsters. Illicit actors will form ties with other actors who play different roles [3], resulting in heterophilic characteristics.

Table 5. Final model hyperparameters.

Dataset	H%	Clases	Nodes	Edges	Features	α	Dropout	λ	Layers	Varient	WD
Cora	0.81	7	2,708	5,429	1,433	0.3	0.6	0.2	2	Eq 5	1e–4
Citeseer	0.74	6	3,327	4,732	3,703	0.5	0.6	1	1	Eq 9	1e–6
Pubmed	0.80	3	19,717	44,338	500	0.2	0.3	1	2	Eq 5	5e–5
Chameleon	0.23	4	2,277	36,101	2,325	0.1	0.3	0.2	1	Eq 5	5e–4
Cornell	0.30	5	183	295	1,703	0.1	0.5	1	1	Eq 9	5e–4
Texas	0.11	5	183	309	1,703	0.3	0.6	1.5	1	Eq 9	5e–4
Wisconsin	0.21	5	251	499	1,703	0.4	0.3	0.2	1	Eq 9	5e–4
PPI		121	56,944	818,716	50	0.5	0.2	1.0	7	Eq 5	0.0

5 Conclusion

We propose FDGATII, a novel efficient dynamic attention-based model that combines attentional aggregation with dual feature preserving mechanisms based on Initial residual and Identity. FDGATII successfully combines strengths of both GCN and GAT worlds with none of the drawbacks, is inductive, able to handle noise in graphs and achieves depths of upto 32; a first for any attentional model and a limitation of many prior GNN techniques. Extensive experiments on a wide spectrum of benchmark datasets show that FDGATII achieves SOTA or second-best accuracy on benchmark fully supervised tasks. FDGATII has exceptional accuracy and efficiency whilst *simultaneously* addressing over-smoothing, suspended animation and heterophily prevalent in real world datasets.

Acknowledegments. Dedicated to Sugandi.

References

1. Abu-El-Haija, S., et al.: Mixhop: higher-order graph convolutional architectures via sparsified neighborhood mixing. In: International Conference on Machine Learning, pp. 21–29. PMLR (2019)
2. Alon, U., Yahav, E.: On the bottleneck of graph neural networks and its practical implications. In: International Conference on Learning Representations (2020)
3. Bright, D., Koskinen, J., Malm, A.: Illicit network dynamics: the formation and evolution of a drug trafficking network. J. Quant. Criminol. **35**(2), 237–258 (2019)
4. Brody, S., Alon, U., Yahav, E.: How attentive are graph attention networks? In: International Conference on Learning Representations (2021)
5. Chen, M., Wei, Z., Huang, Z., Ding, B., Li, Y.: Simple and deep graph convolutional networks. In: International Conference on Machine Learning, pp. 1725–1735. PMLR (2020)
6. Chien, E., Peng, J., Li, P., Milenkovic, O.: Adaptive universal generalized pagerank graph neural network. In: International Conference on Learning Representations (2020)
7. Chitra, U., Musco, C.: Analyzing the impact of filter bubbles on social network polarization. In: Proceedings of the 13th International Conference on Web Search and Data Mining, pp. 115–123 (2020)
8. Hamilton, W.L., Ying, R., Leskovec, J.: Inductive representation learning on large graphs. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, pp. 1025–1035 (2017)
9. Hammond, D.K., Vandergheynst, P., Gribonval, R.: Wavelets on graphs via spectral graph theory. Appl. Comput. Harmon. Anal. **30**(2), 129–150 (2011)
10. Hardt, M., Ma, T.: Identity matters in deep learning. In: International Conference on Learning Representations (2017)
11. Jin, W., Derr, T., Wang, Y., Ma, Y., Liu, Z., Tang, J.: Node similarity preserving graph convolutional networks. In: Proceedings of the 14th ACM International Conference on Web Search and Data Mining, pp. 148–156 (2021)
12. Kim, D., Oh, A.: How to find your friendly neighborhood: graph attention design with self-supervision. In: International Conference on Learning Representations (2020)

13. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: J. International Conference on Learning Representations (ICLR 2017) (2016)
14. Knyazev, B., Taylor, G.W., Amer, M.: Understanding attention and generalization in graph neural networks. Adv. Neural Inf. Process. Syst. **32**, 4202–4212 (2019)
15. Kulatilleke, G.K., Portmann, M., Chandra, S.S.: SCGC: Self-supervised contrastive graph clustering. arXiv preprint arXiv:2204.12656 (2022)
16. LeCun, Y., Bengio, Y., et al.: Convolutional networks for images, speech, and time series. Handb. Brain Theory Neural Netw. **3361**(10), 1995 (1995)
17. Liu, M., Wang, Z., Ji, S.: Non-local graph neural networks. IEEE Trans. Pattern Anal. Mach. Intell. (2021)
18. Luong, T., Pham, H., Manning, C.D.: Effective approaches to attention-based neural machine translation. In: EMNLP (2015)
19. Maurya, S.K., Liu, X., Murata, T.: Simplifying approach to node classification in graph neural networks. J. Comput. Sci. 101695 (2022)
20. Oono, K., Suzuki, T.: Graph neural networks exponentially lose expressive power for node classification. In: International Conference on Learning Representations (2019)
21. Pei, H., Wei, B., Chang, K.C.C., Lei, Y., Yang, B.: Geom-gcn: geometric graph convolutional networks. In: International Conference on Learning Representations, pp. 6519–6528 (2019)
22. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: online learning of social representations. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 701–710 (2014)
23. Rong, Y., Huang, W., Xu, T., Huang, J.: Dropedge: Towards deep graph convolutional networks on node classification. In: International Conference on Learning Representations (2019)
24. Vaswani, A., et al.: Attention is all you need. Adv. Neural Inf. Process. Syst. 5998–6008 (2017)
25. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. In: International Conference on Learning Representations (2018)
26. Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., Weinberger, K.: Simplifying graph convolutional networks. In: International Conference on Machine Learning, pp. 6861–6871. PMLR (2019)
27. Xu, K., Li, C., Tian, Y., Sonobe, T., Kawarabayashi, K.i., Jegelka, S.: Representation learning on graphs with jumping knowledge networks. In: International Conference on Machine Learning, pp. 5453–5462. PMLR (2018)
28. Zhou, J., et al.: Graph neural networks: a review of methods and applications. AI Open **1**, 57–81 (2020)
29. Zhu, J., Yan, Y., Zhao, L., Heimann, M., Akoglu, L., Koutra, D.: Beyond homophily in graph neural networks: Current limitations and effective designs. Adv. Neural Inf. Process. Syst. (2020)