Vol.2021-HPC-181 No.1
2021/9/27

大規模グラフニューラルネットワークにおける実行性能とモデル精度のトレードオフに関する評価

金刺 宏樹[1,a)]　鈴村 豊太郎[2]　劉 欣[1]　広渕 崇宏[1]　高野 了成[1]

概要：グラフデータの大規模化とグラフ構造を用いた機械学習アプリケーションが多様化するにつれて，グラフニューラルネットワーク（GNN）は他の深層学習モデルと同様，GPU や大容量 DRAM などより高性能な計算環境が必要とされている．しかし，グラフデータサイズと比較して GNN の各層のサイズ，学習率などが適切でない場合，GPU デバイスメモリ容量が不足するか，モデル精度が限界になるなどの問題が起こる．本研究では，代表的な大規模グラフデータ，GNN モデルおよび学習用ハイパーパラメータについて，学習時間モデル精度の推移を比較することにより，実行性能とモデル精度のトレードオフを明らかにしながら最適な実行条件を探索する．評価実験により，GNN モデルとそれぞれのハイパーパラメータの違いにより学習開始時から収束するまでの数秒間で顕著な差が見られた．

1. はじめに

近年では，ソーシャルネットワーク，金融取引ネットワークなど頂点数，エッジ数が数百万から数億を超えたグラフに対し，機械学習によって個々のユーザの嗜好予測，潜在的な取引不正などの異常検知を効率化する需要が増えている．これらのグラフ処理と機械学習によって実世界のグラフデータを処理するアプリケーションでは，様々な実験条件，限られた計算資源でなるべく多くのグラフデータを捌かなければならず，モデルを学習させて結果を出すまでの実行時間も限られている．そのため，最高の精度に達していなくても許容できる精度が出れば短時間で学習を打ち切るといった運用が考えられる．特にグラフ構造自体が頻繁に更新されるために，学習と予測のプロセスを数秒以内で完了させる必要のあるソーシャルネットワーク，金融取引ネットワーク，道路交通ネットワークなどにおいて，限られた時間で高精度を実現するようなより実用的なパラメータセットを探索することは重要である．

これらのグラフ構造を考慮した機械学習モデルとして，深層学習をグラフ構造データ向けに応用したグラフニューラルネットワーク (GNN) がある．近年多くの GNN モデルが論文やソースコードとして提案・公開されており，評価用の数百万頂点，数千万エッジ以上の大規模なデータセットについても， Open Graph Benchmark (OGB) [5] などのベンチマークとともに公開されている．これらのモデルの優位性に関する検証実験においては，論文引用ネットワークの Planetoid[14]，ウェブ掲示板投稿ネットワークの Reddit など頂点数が数千〜数十万程度の従来から使われている比較的小規模なデータセットが主に使用されている．また，GNN モデルやベンチマークで採用される評価指標も，学習結果のラベルの正解率や F1-score など，GNN モデルの出力に基づく学習精度のみに限定されている場合が多い．

しかし実社会のアプリケーションでは，上記のように数百万から数億頂点・エッジのグラフデータに対して機械学習を行う需要があり，小規模のグラフデータと同様の結果になるとは限らない．さらに，一般的に GNN の学習時間とそのモデル精度はトレードオフの関係にあるにもかかわらず，公開されている GNN モデルの論文，ベンチマークなどの性能評価対象となるのは後者のモデル精度のみであり，実行時間が考慮されることはほとんどない．

本研究では，論文で提案されている GNN モデルの評価と実際のアプリケーション，グラフデータとのギャップを埋めるために，実行時の性能とモデルの精度について議論する．実世界グラフデータ，GNN モデル，および GNN 学習のハイパーパラメータそれぞれについて，学習時間とモデル精度の推移を評価実験によって比較し，その傾向について議論する．性能と精度のトレードオフを考慮しつつ，決められた実行時間と計算資源で最も良いモデル精度を出すためのグラフデータ，GNN モデル，ハイパーパラメータの関係性を実験によって検証する．

[1] 産業技術総合研究所 情報・人間工学領域
[2] 東京大学大学院 情報理工学系研究科
[a)] kanezashi.h@aist.go.jp

本書の構成は以下の通りである．第 2 章では GNN のモデルおよびハイパーパラメータ，計算機環境をもとに，現状の GNN の学習に関する課題について述べ，第 3 章では大規模グラフデータセットを用いた実世界アプリケーションの動向と需要について，代表的な事例やベンチマークなどを参照しながら説明する．第 4 章では，代表的な GNN モデルとベンチマークとして用いられる大規模グラフデータを使用して，ハイパーパラメータを変えながら性能評価を行い，実行性能とモデル精度の関係性をそれぞれ比較する．第 5 章では GNN の性能評価に関する関連研究について説明し，第 6 章で本研究のまとめと今後の展望について述べる．

2. GNN モデル評価に関する問題提起

本章では，実世界で用いられているグラフデータとアプリケーションの現状と，このようなグラフデータを GNN モデルに適用させる際に想定される必要な指標と制約条件について述べる．具体的なデータセット，機械学習が必要ないくつかのシナリオについて，それに対処するための GNN モデル，計算機環境をそれぞれ取り上げながら，どのような指標を最適化すれば良いか議論する．

数百万頂点を超える大規模かつ動的に構造が変化する実世界グラフデータに対応するために，GCN モデルに LSTM, GRU など時間的変化を考慮したユニットを組み込んだ動的な GNN モデルや，GCN モデルを簡略化し時間的・空間的計算量をそれぞれ削減することで大規模グラフを一度に扱えるようにした SGC[13] などの派生モデルも数多く提案されている．

これらの GCN 派生モデルを従来のモデルと比較する際に評価用として使われていたグラフデータは，部分的な論文引用ネットワークのスナップショットなど比較的小規模な静的データセットに限定されており，Cora, Citeseer, Pubmed などの Planetoid [14] に代表される数千〜数万頂点・エッジ規模の固定されたデータセットが主に使われていた．しかし実際の運用では数百万から数億頂点・エッジ以上のグラフデータを一度に扱う必要があり，同様の GNN モデル，ハイパーパラメータ，ハードウェアの条件で動かすことは不可能である．仮に同様の条件でさらに大規模なグラフデータを扱えた場合であっても，実行性能の違いにより異なるハイパーパラメータセットが最適となる場合もあり，そもそも学習に要する時間が長くなることで実用に耐えない恐れもある．

より大規模で実践的なグラフデータを前提として評価できることを目的に，最近ではこれらの大規模グラフを想定した Benchmarking-GNNs [1], Open Graph Benchmark (OGB)[5] などの GNN 用ベンチマーク，および OGB が主催する OGB-LSC KDD Cup 2021 [4] などのコンペティションが公開されている．しかし，これらのコンペティションでは，基本的に最終的なモデル精度のみが評価の対象となり，モデルの学習時間に関する制約は少ない．一般的に学習に要した時間とモデル精度はトレードオフの関係にあり，モデル精度を向上させるためには GPU など高性能なハードウェアを長時間使用する必要がある．ただし，実世界の動的グラフにおいては，前述のような異常検知アプリケーションなどをより精緻にするために，データの更新と学習の間隔は短いほど好ましいため，限られた時間や計算資源では GNN モデルが収束する前に学習を打ち切りデータを更新しなければならない場合もある．

3. GNN モデルとハイパーパラメータ

本章では，代表的な GNN モデルの種類と，付随するハイパーパラメータごとに性能面での特徴を述べる．特に，学習に要する計算コストと全体的なモデル精度の双方について議論する．

3.1 代表的な GNN モデル

深層学習モデルの中で基本的な Convolutional Neural Network (CNN) をグラフ構造に適用させた Graph Convolutional Network (GCN)[7] をはじめとした，数多くの派生モデルが提案されている．ここでは，本研究で評価対象とする代表的な GNN モデルである GCN, GraphSAGE[3], SGC (Simple GCN)[13], GAT (Graph Attention Network)[11] の概要について説明する．

GCN [7] 近傍のネットワークに畳み込み処理を適用させた GNN モデルであり，各頂点を表現するベクトル (畳み込みベクトル) の内容を，重み付けを行いながら隣接頂点に伝播させていく．

GraphSAGE [3] GCN モデルでは各頂点が全ての隣接頂点に相当する畳み込みベクトルを伝搬させるが，計算対象とする頂点をサンプリングし周辺の畳み込みベクトルを集約させて学習することで，時間的計算量を削減しつつグラフ構造による表現を効率的に学習する．

SGC [13] GCN モデルの中から，GNN 計算で本質的に不要な非線形関数を除去し，学習対象の重み行列を集約させることで計算量を軽減する．GNN 構造が簡略化されたことで，後述するハイパーパラメータの隠れユニット数，ドロップアウト率は適用されない．

GAT [11] 各ユニット内のパラメータに加え，Attention と呼ばれる隣接頂点間の頂点同士を結ぶエッジの重要度も同時に学習することで，特徴ベクトルの伝搬に重み付けを加え，より精緻なモデルの学習を可能にしている．GCN モデルよりも学習の自由度は高くなるが，エッジ数に比例した Attention のパラメータも学習するため，時間的・空間的計算量が増大する．

図 1 は GNN モデルの概要である．この図で下線を引いた学習率 r，隠れ層数 k，隠れユニット数 n，ドロップアウ

ト率 d が本研究で検証するハイパーパラメータである．

3.2 GNN のハイパーパラメータ

GNN のモデル精度を左右するのはモデルの種類だけではなく，学習率，隠れ層のサイズなど GNN の構造を決定するハイパーパラメータも含まれる．一般的な傾向が次に述べる通りであるが，実際の実行速度やモデル精度への影響は使用するハードウェア，グラフデータのサイズや形状によって異なる．

3.2.1 学習率

学習率（Learning Rate）は，他のニューラルネットワークモデルの学習と同様，誤差逆伝播法により予測結果と教師データとの誤差を GNN モデルにフィードバックする際の重み付け用ハイパーパラメータである．学習率が大きいほどより少ないエポック数で短時間で最適解に達することが多いが，大きすぎると学習したモデルが不安定になり，最適解から遠ざかることもある．

3.2.2 隠れ層・隠れユニット数

隠れ層（中間層）も，一般的なニューラルネットワークと同様に，その数が多いほどより精緻なモデルを表現可能である一方，モデル更新のための行列計算を繰り返す必要があり，その分時間的，空間的計算量が必要とされる．GCN モデルにおいては，隠れ層を一つ増やすことにより，頂点自身が持つ畳み込みベクトルをさらに 1 ホップ隣の頂点へ伝搬させることができ，さらにグラフ構造を考慮した学習が可能となる．上記の隠れ層と同様に，隠れユニット数，つまり隠れ層あたりの学習パラメータの個数によってもニューラルネットワークの表現力は変化する．なお SGC モデルでは隠れ層に相当する計算を行列積計算に簡略化させたため，隠れユニットに相当するハイパーパラメータは存在しないが，隠れ層数は隣接頂点のホップ数に相当するため，以下の評価実験では同様に扱う．

3.2.3 ドロップアウト率

ニューラルネットワークモデル精度を向上させるための重要な手法として，ドロップアウト [10] がある．ドロップアウトでは，一定割合の隠れユニットを学習時にランダムに無効にすることで，学習データに特化する過学習を抑制する．ドロップアウト率が高いほど，学習対象となる隠れユニットの割合が小さくなるが，GNN 全体の学習により多くの学習データと時間を要するトレードオフがある．

4. GNN モデルの学習時間と精度の推移評価

2 章で述べた通り，ソーシャルネットワーク，金融取引ネットワークなど頂点数，エッジ数がそれぞれ数百万，数億を超えるグラフデータをそのまま GCN モデルで扱うことは困難であり，頂点のサンプリング，グラフクラスタリングによって一度に学習する頂点セットを限定するなどの近似手法が主流となっている．一方で，近似手法を使用したことによるモデル精度が悪化したり，モデルが収束するまで長時間を要することもある．

本章では，2, 3 章で述べた内容を踏まえ，グラフデータセット，GNN モデル，およびハイパーパラメータの組み合わせの違いにより，学習時間に対するモデル精度の推移を評価する．評価実験の結果をもとに，実行時間が制限されたと仮定した場合にどのような組み合わせが最適化を考察する．

4.1 実験環境と条件

4.1.1 ハードウェア・ソフトウェア構成

本実験では，GNN モデルを学習させるハードウェアとして計算機サーバ NVIDIA DGX-1 [8] を使用した．DGX-1 に搭載されているプロセッサとメインメモリ の概要は，それぞれ表 1 の通りである．それぞれの GNN 学習プロセスに対し，GPU デバイスを一台ずつ使用している．

表 1 DGX-1 の概要

GPU	8X NVIDIA® Tesla® V100, 16GB
CPU	Intel® Xeon® CPU E5-2698 v4 2.20GHz, 2 x 20 cores
DRAM	512 GB 2,133 MHz DDR4 RDIMM

また，本実験では DGX-1 マシンの中に CUDA が利用可能な NVIDIA Docker コンテナを導入し，その中で性能評価を行った．CUDA や Python パッケージなどソフトウェアの概要は以下の表 2 の通りである．また，GNN モデルを設計・実行するためのフレームワークとして，PyTorch バックエンドの PyTorch Geometric (PyG) [2] を使用した．

表 2 ソフトウェアの概要

Host OS	Ubuntu 18.04.4 LTS
Docker	19.03.15, build 99e3ed8919
GPU Driver	418.197.02
CUDA	11.0
GCC	7.5.0
Python	3.6.9
PyTorch	1.7.1+cu110
PyTorch Geometric	1.6.3

4.1.2 比較対象の GNN モデルとハイパーパラメータ

本実験では，GNN モデルとして GCN, GraphSAGE, SGC, GAT をそれぞれ使用した．また，本実験で評価対象としたハイパーパラメータの候補は以下の表 3 の通りである．太字はデフォルトのパラメータ値であり，前述の Open Graph Benchmark [5] で公開されているサンプルプログラムを参考にした．

基本的にニューラルネットワーク学習において，同一のグラフデータを繰り返し使用する学習回数（エポック数）を増やすほどモデル精度が上がる傾向にあるが，エポック数に概ね比例した学習時間がかかり，GNN 内部の学習パ

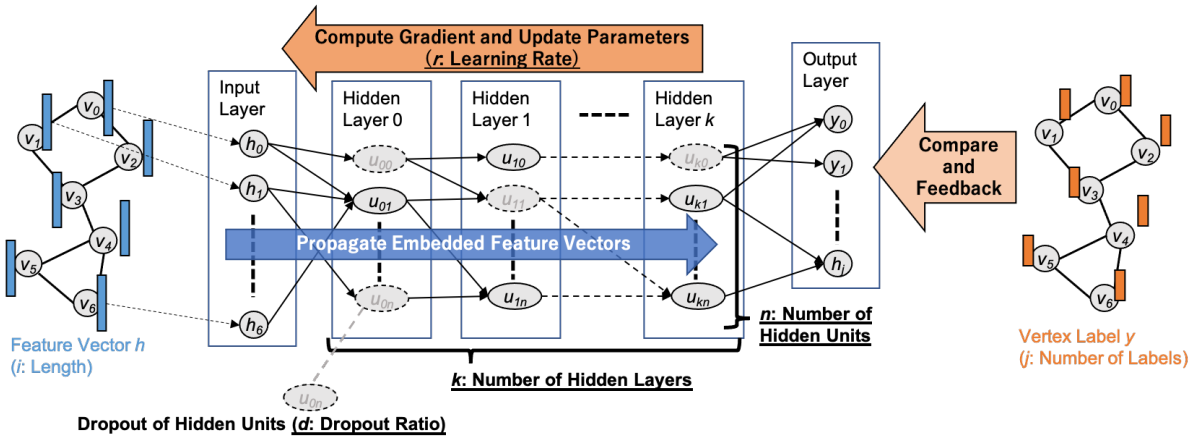

図 1 GNN の構造とハイパーパラメータ

表 3 ハイパーパラメータの候補

学習率 r	0.001, 0.002, 0.005, **0.01**, 0.02, 0.05
隠れ層数 k	1, **2**, 3, 4
隠れ層ユニット数 n	16, 32, **64**, 128, 256, 512
ドロップアウト率 d	0.00, 0.25, **0.50**, 0.75

ラメータの変化がほとんどなくなりモデルが収束することで，モデル精度が頭打ちになる．モデルの収束を判定して学習を途中で打ち切る場合，基本的にニューラルネットワーク内部の学習パラメータの変化量を指標としているが，本研究の実験では，エポック数の上限を 1,000 (SGC モデルでは 2,000) とした上で，加えて ogbn-arxiv データセットでは実行開始から合計 60 秒，ogbn-mag, ogbn-products データセットでは 180 秒経過した時点で学習を打ち切るようにした．

4.1.3 グラフデータセット

本実験で使用するグラフデータセットは，Open Graph Benchmark [5] で公開されている Node Property Prediction カテゴリから，論文引用ネットワーク ogbn-arxiv, ogbn-mag, 購買ネットワーク ogbn-products データセット [9] を使用した．それぞれのグラフデータの統計情報は表 4 の通りである．ogbn-arxiv, ogbn-mag データセットの頂点の特徴量は，いずれも論文のタイトルとアブストラクトに含まれる単語ベクトルを 128 次元に畳み込んだものであり，同様に ogbn-products の頂点の特徴量も商品の説明文のテキストデータを 100 次元のベクトルに畳み込んだものである．この特徴量数とラベル数は，それぞれ図 1 で示す左端，右端の特徴量ベクトルの長さ i と頂点ラベルの種類数 j に相当する．

表 4 使用したグラフデータセット

データセット名	頂点数	エッジ数	特徴量数	ラベル数
ogbn-arxiv	169,343	1,166,243	128	40
ogbn-mag	1,939,743	21,111,007	128	349
ogbn-products	2,449,029	61,859,140	100	47

4.2 GNN モデルによる性能傾向

ogbn-arxiv データセットに対して，GCN, GraphSAGE, SGC, GAT モデルそれぞれの学習時間とモデル精度の傾向は図 2 のようになった．なお，GPU メモリ制限の問題から，GAT モデルは ogbn-arxiv データセットのみに対して学習できた．

最終的なモデル精度は SGC を除きいずれの GNN モデルも 69 - 70% 前後で推移しているが，学習開始時から 5 秒前後で GCN, GraphSAGE モデルはほぼ最高のモデル精度に達しているのに対し，SGC, GAT モデルは比較的モデルの学習が緩やかであり，学習開始から 20 秒近く経過しないと GCN に並ぶ精度に及ばなかった．それぞれのモデルの学習開始時から 1, 2, 5, 10 秒経過した時点と，学習終了時のモデル精度は表 5 の通りである．特に学習開始から 1, 2 秒経過した時点において，SGC, GAT モデルの精度向上が著しく低いことが分かる．

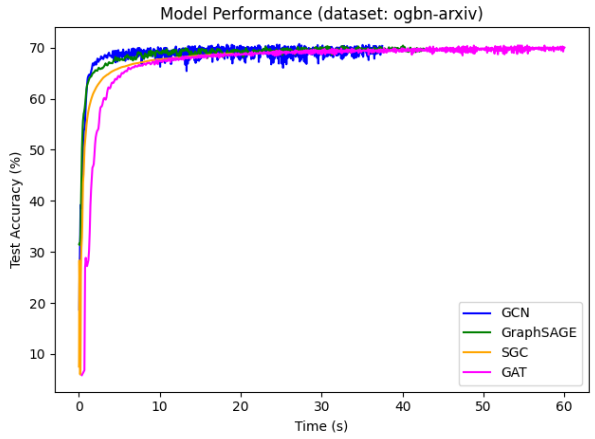

図 2 GNN モデル別の学習時間と精度の推移 (ogbn-arxiv)

図 3 の ogbn-mag データセット，図 4 の ogbn-products の結果では，GraphSAGE と SGC モデルにおいてはほぼ同様の傾向であったが，GCN では学習モデルが収束せず，一貫して不安定な結果となった．ogbn-mag データセット

GNN モデル	1 s	2 s	5 s	10 s	Final
GCN	62.4%	67.2%	68.0%	67.9%	70.3%
GraphSAGE	62.6%	65.4%	67.5%	69.0%	69.8%
SGC	55.4%	61.7%	66.0%	67.7%	68.8%
GAT	27.2%	51.1%	64.2%	67.3%	70.0%

表 5 GNN モデル別の精度の途中経過 (ogbn-arxiv)

では精度が 2% 前後にとどまり，ogbn-products では学習開始から 10 秒でピークの精度 64.5%に達した後精度が下がり続け，80 秒程度で SGC の精度 60.4%を下回る結果になった．GCN のモデル精度が向上しない原因については今後詳細に調査する予定である．

また，SGC モデルは [13] において短時間で GCN に匹敵するモデル精度を実現するとあるが，SGC モデルの精度を十分向上させるためには多くのエポック数を要するため，学習時間を基準にモデル精度向上の過程を比較した場合，GraphSAGE と傾向はほぼ同じか，やや低い傾向にあることが分かった．

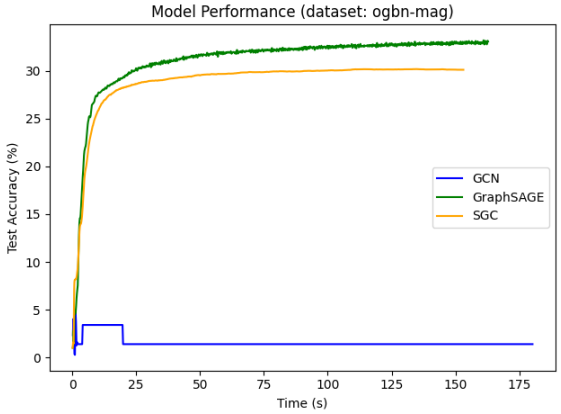

図 3 GNN モデル別の学習時間と精度の推移 (ogbn-mag)

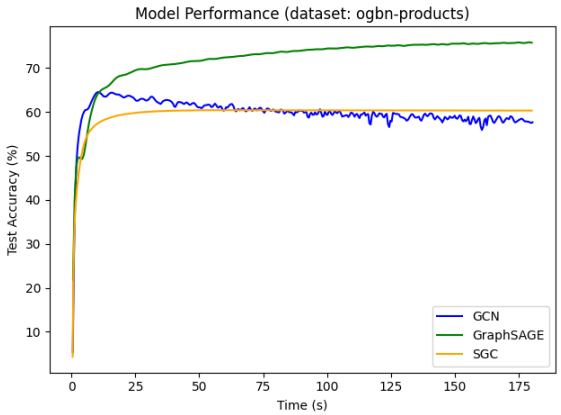

図 4 GNN モデル別の学習時間と精度の推移 (ogbn-products)

その他のハイパーパラメータセットでもこの 4 種類のモデルについて同様の比較実験を行ったが，GAT モデルは GPU メモリの不足により ogbn-mag, ogbn-products データセットを扱うことができず，また GCN モデルは GraphSAGE, SGC モデルと比較して上記の通りモデル精度が向上しなかったため，以下 GraphSAGE と SGC モデルそれぞれについて比較実験を行った．

4.3 学習率による性能傾向

学習率の違いによる，GraphSAGE と SGC モデルそれぞれの実行時間に対するモデル精度の傾向は図 5, 6 のようになった．GraphSAGE モデルにおいては，1,000 エポックまで学習した場合のモデル精度はいずれもほぼ同じであるが，0.002 以下 の極端に低い学習率では特に開始 10 秒まではモデルの精度向上が比較的遅いことが分かる．SGC モデルでは学習率による違いがさらに顕著になっており，最終的なモデル精度でも学習率 0.001, 0.050 でそれぞれ 62.6%, 69.1%と明確な差が出ている．

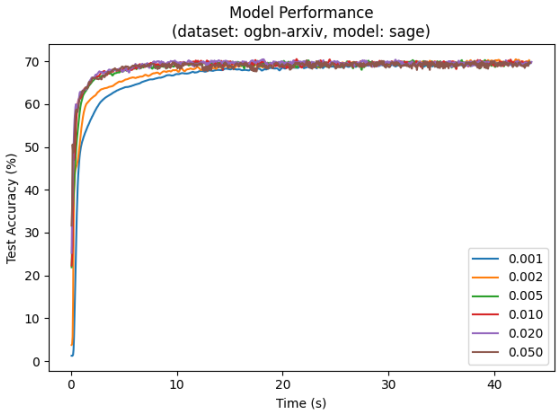

図 5 学習率による GraphSAGE モデルの精度推移

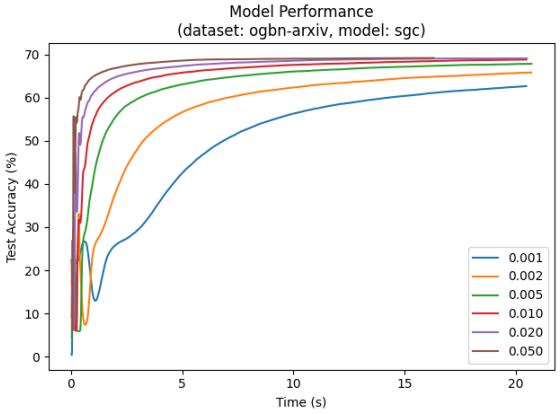

図 6 学習率による SGC モデルの精度推移

同様の条件で，ogbn-mag データセットで GraphSAGE, SGC モデルを学習させた結果は図 7, 8 の通りになった．GraphSAGE モデルでは学習率が 0.020 の場合に最もモデル精度が高い傾向にあり，1,000 エポック学習後は精度が 33.1% に達した．しかし，さらに高い 0.050 では学習開始から 15 秒程度で 28%まで達したものの，それ以降は精度が不安定となり，最終的に最も学習率の低い 0.001 の場合よりもさらに低い精度となった．一方，SGC モデルでは一貫して学習率が高いほどモデル精度が上がりやすくなる

傾向が顕著であり，学習率が 0.050 の場合はわずか 8.1 秒で 29% まで向上した．これは同条件の GraphSAGE モデルでは達成できていない結果である．

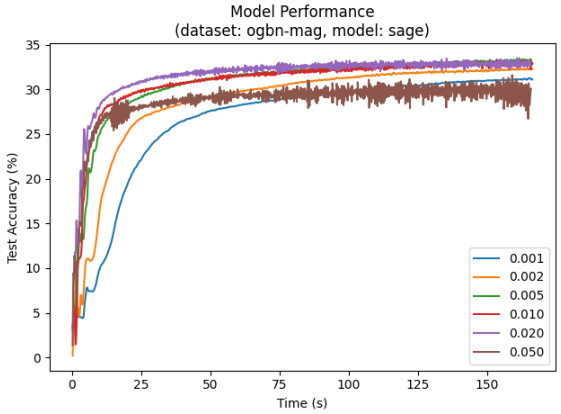

図 7 学習率による GraphSAGE モデルの精度推移

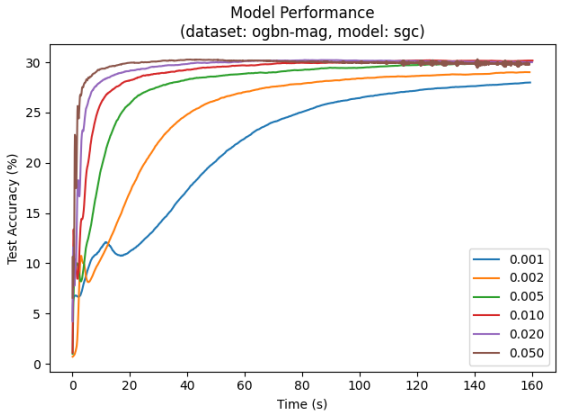

図 8 学習率による SGC モデルの精度推移

4.4 隠れ層数による性能傾向

GraphSAGE モデルの隠れ層数，SGC モデルの隣接頂点ホップ数におけるそれぞれの実行時間に対するモデル精度の傾向は図 9, 10 の通りである．GraphSAGE モデルにおいては，いずれの隠れ層数でも 20 秒以内に精度が 70% 近くまで到達したが，学習の序盤でのモデル精度向上が遅く，隠れ層数にほぼ比例して全体の学習時間が長くなる結果になった．SGC モデルでは GraphSAGE と比較してモデル精度の推移が安定しており，ホップ数が 2 以上ではほとんど変化が見られなかった．

4.5 隠れ層ごとのユニット数による性能傾向

GraphSAGE モデルの隠れユニット数（隠れ層サイズ）におけるそれぞれの実行時間に対するモデル精度の傾向は図 11 の通りである．隠れ層数での傾向と同様，いずれの隠れユニット数でも 20 秒以内に精度が 70%近くまで到達したが，特に隠れ層数が多い場合に，図 9 と同様に序盤のモデル精度向上が鈍くなっている．いずれも最終的なモデル精度は 70%とほぼ同じだが，学習しなければならないパラメータが隠れ層数，隠れユニット数に比例して増大するため，特に短時間での学習ではこれらのパラメータを小さく設定することが有効と言える．

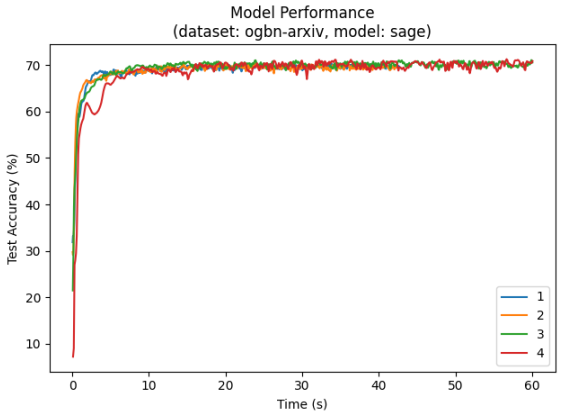

図 9 隠れ層数による GraphSAGE モデルの精度推移

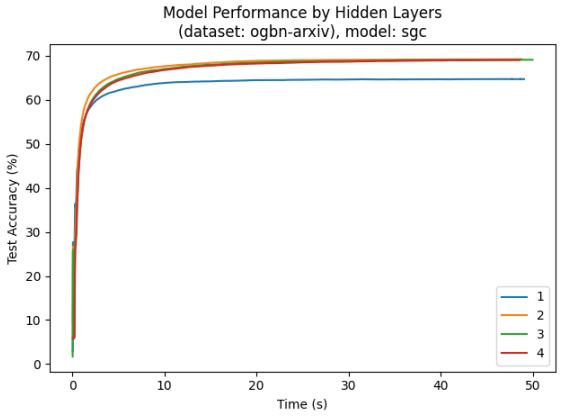

図 10 ホップ数による SGC モデルの精度推移

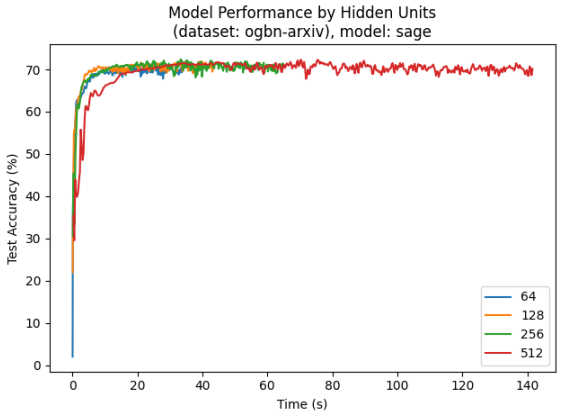

図 11 隠れユニット数による GraphSAGE モデルの精度推移

4.6 ドロップアウト率ごとの性能傾向

図 12, 13, 14 は，それぞれ ogbn-arxiv, ogbn-mag, ogbn-products データセットでドロップアウト率を変化させた場合のモデル精度の変化を示したものである．なお，SGC

モデルでは GNN モデルの構造上ドロップアウトを扱わないため，GraphSAGE のみの結果を示す．

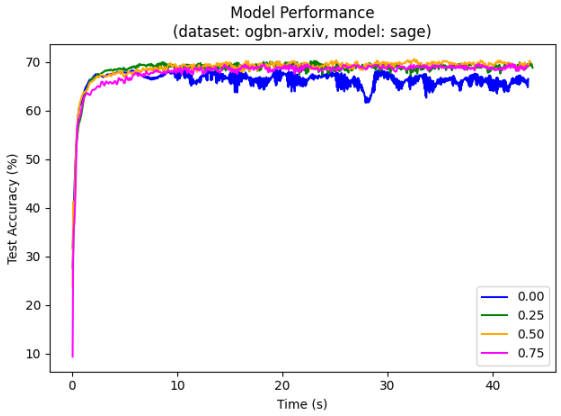

図 12 ドロップアウト率による GraphSAGE モデルの精度推移 (ogbn-arxiv)

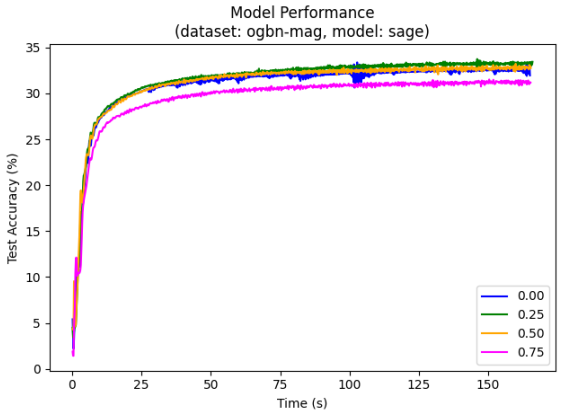

図 13 ドロップアウト率による GraphSAGE モデルの精度推移 (ogbn-mag)

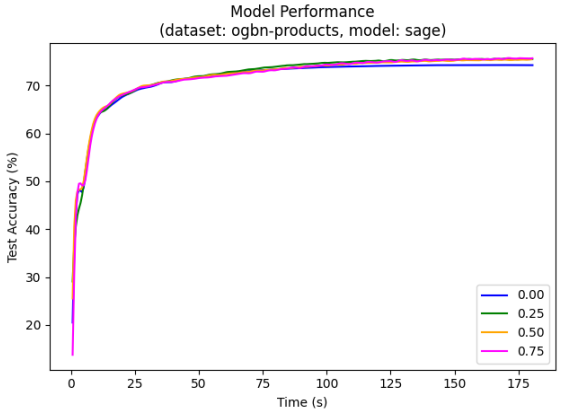

図 14 ドロップアウト率による GraphSAGE モデルの精度推移 (ogbn-products)

比較的データセットの小規模な ogbn-arxiv データセットでは，ドロップアウト率が 0.0，すなわち全くドロップアウトを適用しなかった場合にモデル精度が不安定となった一方，ドロップアウト率を 0.25 にすると学習開始から 3.0 秒で 68%の精度に至った．また，ドロップアウト率が高めの 0.75 ではモデル精度の向上が他の条件と比較して緩やかであり，同じ 68%の精度に達するまでに 8.0 秒要した．

ogbn-mag データセットでは，ドロップアウト率が 0.25, 0.5, 0.0 の順にモデル精度が高い傾向にあり，ドロップアウト率が 0.75 の実際の 4 分の 1 の隠れユニットしか学習しない場合では 1,000 エポックで学習を終えた時点で 2.2%低い制度となった．このデータセットにおいても，ドロップアウトを行わなかった場合に不安定な学習結果が見られる．これらの結果から，短時間で学習を打ち切る場合は，ドロップアウト率を低めに設定し，GNN 学習を進めるのが望ましいと分かった．ogbn-products では，ドロップアウトを行う場合と全く行わない場合でタイムアウト時の多少の差はあるものの，いずれのドロップアウト率でも安定してモデル精度が向上していた．

5. 関連研究

代表的な GNN モデルとそのハイパーパラメータを網羅的に評価したサーベイ論文はある一方，その評価指標はモデル精度に特化したものが多く，学習時間などランタイムの性能を考慮した研究は少ない．[6] では，予測した GNN の実行性能と現実の実行時間の乖離の原因について不規則なメモリアクセスを考察し，本研究で使用した PyG の他，Deep Graph Library (DGL) などの既存のフレームワークを相対的な実行時間，GCN 内部の計算スループット，プロセッサのキャッシュヒット率によって比較している．[12] では，GNN の内部構造に着目し，学習プロセスを頂点ごとに処理される vertex calculation とエッジごとに処理される edge calculation に分け，GNN の性能ボトルネックについて学習プロセス別に詳細に考察している．評価指標としては 1 エポックごとの平均学習時間，GNN モデル学習時の最大メモリ使用量を使用しているが，それぞれミニバッチサイズの比較に関してのみ精度を使用しており，実行時の精度比較は行われていない．

本研究では，GNN の学習時間とモデル精度のトレードオフを考慮し，エポック数ではなく全体の学習時間に対するモデル精度の評価を行い，特に学習開始時から数秒間のモデル収束前における精度を比較することにより，実世界アプリケーションで運用する際に有用となる評価実験を行った．

6. 結論と今後の展望

本研究では，代表的な GNN モデルとそのハイパーパラメータについて，実行時間とモデル精度双方の観点から性能の比較と傾向を評価した．GNN モデルにより最終的なモデル精度に違いがない場合であっても，モデルが安定するために要する時間，学習開始時から数秒後のモデル精度に明らかな違いが見られた．特に SGC モデルについては，我々の実験では十分にモデル精度が向上するまで多

くのエポック数を要したため，学習時間を基準とするとGraphSAGE と比較して優位性は見られなかった．また，学習率，隠れ層数，隠れユニット数においても実行時間によるモデル精度向上の過程に違いがみられ，短時間での学習において最適なハイパーパラメータの指針を示すことができた．

今回は代表的な GNN モデルと OGB データセットの一部を用いて性能評価を行なったが，実世界データセットとアプリケーションに対応する評価としてはまだ不十分である．まず，今回使用したグラフデータセットの規模は百万頂点，一千万エッジを超えるものであるが，実際のソーシャルネットワーク，金融取引ネットワークなどは数億エッジ規模であるため，さらに大規模なグラフデータによる検証が必要である．また，グラフサイズだけでなく，構造が時系列で変化する動的グラフ，異なる属性の頂点，エッジが混在するヘテロジニアスグラフもこれらの実世界アプリケーションでは用いられるため，同様のグラフデータとサポートする GNN モデルによる性能評価も進める予定である．さらに，GNN モデルを実際に運用することを考慮すると，GNN モデルの種類，グラフデータセットに加え，実行環境の違いによる学習時間とモデル精度の評価も重要である．今回はシングル GPU による性能評価実験を行ったが，利用可能な GPU メモリ容量の制限により GAT モデルが扱えなかったため，さらに大規模なグラフを扱うことを鑑みてマルチ GPU かつ分散環境上においても同様の評価を行う予定である．

謝辞 本研究の一部は，JSPS 科研費 JP21K17749 の助成を受けたものである．

参考文献

[1] Dwivedi, V. P., Joshi, C. K., Laurent, T., Bengio, Y. and Bresson, X.: Benchmarking graph neural networks, *arXiv preprint arXiv:2003.00982*.
[2] Fey, M. and Lenssen, J. E.: Fast Graph Representation Learning with PyTorch Geometric, *ICLR Workshop on Representation Learning on Graphs and Manifolds* (2019).
[3] Hamilton, W. L., Ying, R. and Leskovec, J.: Inductive representation learning on large graphs, *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 1025–1035 (2017).
[4] Hu, W., Fey, M., Ren, H., Nakata, M., Dong, Y. and Leskovec, J.: OGB-LSC: A Large-Scale Challenge for Machine Learning on Graphs, *arXiv preprint arXiv:2103.09430* (2021).
[5] Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., Catasta, M. and Leskovec, J.: Open graph benchmark: Datasets for machine learning on graphs, *arXiv preprint arXiv:2005.00687* (2020).
[6] Huang, K., Zhai, J., Zheng, Z., Yi, Y. and Shen, X.: Understanding and Bridging the Gaps in Current GNN Performance Optimizations, New York, NY, USA, Association for Computing Machinery (2021).
[7] Kipf, T. N. and Welling, M.: Semi-Supervised Classification with Graph Convolutional Networks, *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.
[8] NVIDIA: AI 研究向けディープラーニング サーバー - NVIDIA DGX-1 (2021).
[9] Open Graph Benchmark: Node Property Prediction (2021).
[10] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting, *The journal of machine learning research*, Vol. 15, No. 1, pp. 1929–1958 (2014).
[11] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P. and Bengio, Y.: Graph Attention Networks, *International Conference on Learning Representations* (2018).
[12] Wang, Z., Wang, Y., Yuan, C., Gu, R. and Huang, Y.: Empirical analysis of performance bottlenecks in graph neural network training and inference with GPUs, *Neurocomputing*, Vol. 446, pp. 165–191 (2021).
[13] Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T. and Weinberger, K.: Simplifying graph convolutional networks, *International conference on machine learning*, PMLR, pp. 6861–6871 (2019).
[14] Yang, Z., Cohen, W. and Salakhudinov, R.: Revisiting semi-supervised learning with graph embeddings, *International conference on machine learning*, PMLR, pp. 40–48 (2016).