修士論文

グラフ構造を用いて欠損値埋めを行うネットワークの半教師あり学習

吉川 純平

20M31190

東京工業大学
情報理工学院
知能情報コース

指導教員　村田 剛志

2020 年 1 月

概要

近年 Twitter や Facebook などの SNS の普及によりネットワーク上でのコミュニケーションが可能になったことや、化合物の物性推定の重要性が高まったなどにより、グラフ構造を含むデータの分析が注目を集めている。グラフとは「ノード」と、二つのノード間を結ぶ「エッジ」から構成されるデータ構造である。従来の研究において、グラフ構造とそれぞれのノードの特徴量を用いて予測する Graph Neural Network(GNN) が優れた成果を残している。その中でも、1 層で距離 1 の隣接ノードの特徴量の畳み込みを行う Graph Convolutional Network(GCN) が、半教師ありノード分類やリンク予測などの様々なタスクで高い精度を示している。

GCN は学習の際に、特徴量に欠損値を含まないことを前提とした手法である。しかし実世界において欠損値を含むデータは数多く存在する。例えば人的ミス、センサーのエラー、任意回答における未入力項目を含むアンケートなどにより欠損値を含むデータを得られることがある。従来の欠損値を含むグラフデータに欠損値補完をする手法の多くは、グラフ構造を無視した機械学習手法により欠損値を穴埋めし、得られたデータを特徴量として GCN などのモデルを学習させる手法が一般的である。この方法は代入法にグラフ構造を用いないため予測精度が低くなる可能性がある。

本研究ではこのような欠損値を含むグラフデータに対して、グラフ構造を用いて欠損値補完を行うことで、GCN を用いた予測精度を向上させる手法を提案する。提案手法はグラフの近接ノードの情報を再帰的に集約し更新する手法を用いて欠損値を補完し、GCN を用いて予測するモデルにより構成されている。提案手法はグラフ構造を用いて欠損値補完を用いることで、GCN を用いて予測するために適した特徴量を扱えるため、予測精度を向上させることができた。

提案手法の有効性を示すために半教師ありノード分類とリンク予測の 2 つのタスクを実験として行った。実験ではノードの特徴量は欠損値の割合を 10% から 90% まで 10% 刻みで変化させて予測精度を調べた。実験の結果、主に欠損率が高い場合に提案手法は既存手法と比べて高い精度を得ることを確認した。

目次

図目次

表目次

第 1 章

序論

グラフ構造を持つデータの分析は近年の重要な研究分野の一つとされている。グラフとは「ノード」と、二つのノード間を結ぶ「エッジ」から構成されるデータ構造である。従来は、グラフ構造を持つデータの解析は情報量の多さやデータの複雑性から多くは研究されていなかった。しかし近年 Twitter や Facebook などの SNS の普及によりネットワーク上でのコミュニケーションが可能になったこと [12] や、化合物の物性推定の重要性が高まったこと [11] などにより、グラフ構造を含むデータの分析が注目を集めている。

このタスクを解決するためにグラフベース正則化 [4] というアプローチを用いた Label Propagation [57] や Label Spreading [55] などの手法が古くに提案された。この手法はグラフのエッジに注目し、隣接ノード同士は等しいラベルを持つ可能性が高いという経験に基づいた手法である。次にグラフエンベディング (graph embedding) というノードやエッジなどをグラフ構造を保持したまま低次元ベクトル空間に埋め込む手法が提案された。グラフエンベディングにはランダムウォークを用いるモデル [35] や、エンベディングした特徴量からグラフを再構成するモデル [47] などがある。

そして第 3 のアプローチとして spectral convolution [6] というグラフ構造に対して畳み込みを演算を行う手法が提案された。この手法は音声のフィルタリングと同様に、グラフ信号に対してフーリエ変換を行うことでグラフ構造での畳み込み演算を行う手法である。この手法の畳み込み演算は数学的な理論に基づいて行うことができるが、計算量が大きくなってしまう問題がある。ニューラルネットワークをグラフに適用し学習する Graph Neural Network(GNN) はその計算量の問題を解決できる手法として提案された。GNN はノードの特徴量だけでなく、グラフ構造を学習することができる。その中でも、1 層で距離 1 の隣接ノードの特徴量の畳み込みを行う Graph Convolutional Network(GCN) [24] が、半教師ありノード分類やリンク予測などの様々なタスクで高い精度を示している。

GCN は学習の際に、特徴量に欠損値を含まないことを前提とした手法である。しかし実世界において欠損値を含むデータは数多く存在する。例えば (1) 人的なミス、(2) センサーによるエラー入力、(3) 任意回答における未入力項目を含むアンケート、(4) ビッグデータの情報欠損などにより、欠損値を含むデータを得られることがある。

欠損値を含むグラフデータを用いて学習する方法は、(1) 欠損値を除去して学習する方法、(2) 代入法を用いて欠損値を補完し、得られた欠損値を含まない特徴量に既存の機械学習方法を用いて学習する方法、(3) 欠損値を含む特徴量をそのまま用いて学習する手法の 3 種類が存在する。従来の代入法の多くは、グラフ構造を無視して機械学習手法により欠損値を穴埋めしているために予測精度が低くなる可能性があるという問題がある。

本研究ではこのような欠損値を含むグラフデータに対して、グラフ構造を用いて欠損値補完を行うことで、GCN を用いた予測精度を向上させる手法を提案する。ノードの欠損値を近隣ノードの特徴量を平均して補完する近隣平均補完を用いた GCN_neighbor モデルと、欠損値を再帰的に補完する近隣再帰補完を用いた GCN_recursive モデルを提案手法とした。提案手法はグラフ構造を用いて欠損値補完

を用いることで、GCN を用いた予測に適した特徴量を扱える強みがある。

提案手法の有効性を示すために半教師ありノード分類とリンク予測の 2 つのタスクを実験として行った。実験ではノードの特徴量は欠損値の割合を 10% から 90% まで 10% 刻みで変化させて予測精度を調べた。実験の結果、主に欠損率が高い場合に、提案手法は既存手法と比べて高い精度を得ることを確認した。また提案手法におけるハイパーパラメータである再帰回数が十分であることを実験を通して検討した。

本論文は全 6 章から構成される。第 2 章では本研究で利用する従来の手法や関連研究について述べる。第 3 章では本研究における提案手法について具体的に述べる。第 4 章では論文引用ネットワークや共購買ネットワークを用いた実験を行う。第 5 章では本論文の結論と今後の課題について述べる。

第 2 章

関連研究

2.1　グラフ構造

2.1.1　グラフ構造の表現

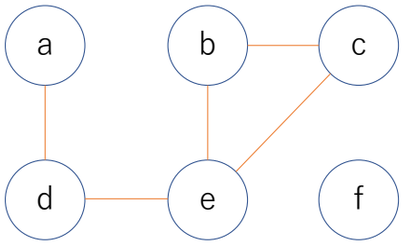
グラフはノードと 2 つのノード同士を結ぶエッジから構成されている。つまりノードの集合を $\mathcal{V}$、エッジの集合を $\mathcal{E}$ とすると、グラフ $\mathcal{G}$ は $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ と表せる。グラフの種類としては、エッジに向きが存在する有向グラフ、同じノードペアに多重のエッジが張られるグラフ、セルフループというノードから同じノードにエッジが張られるグラフや、エッジごとに重みがあるグラフなど様々な種類が存在するが、本研究ではセルフループなしの無向重みなしグラフを用いる。そのようなグラフは隣接行列 $\mathbf{A} \in R^{N \times N}$ を用いて表現することができる。ここでノード v_i, v_j 間にエッジが存在する場合、つまり $(v_i, v_j) \in \mathcal{E}$ のときには $A_{ij} = 1$ として表現する。一方でノード v_i, v_j 間にエッジが存在しない場合、つまり $(v_i, v_j) \notin \mathcal{E}$ のときには $A_{ij} = 0$ として表す。図 2.1 はグラフ構造の例である。この例の左図ではノード数 6、エッジ数 5 のグラフを視覚的に示しており、そのグラフ構造を右図では隣接行列を用いて示している。

2.1.2　グラフ構造を用いたタスク

Getoor [14] や Hamilton [18] によるとグラフ構造を用いたタスクは以下のようなものがある。

- ノードごとのタスク
  - ノードの分類

Graph: $\mathcal{G} = (\mathcal{V}, \mathcal{E})$



隣接行列: A

	a	b	c	d	e	f
a	0	0	0	1	0	0
b	0	0	1	0	1	0
c	0	1	0	0	1	0
d	1	0	0	0	1	0
e	0	1	1	1	0	0
f	0	0	0	0	0	0

図 2.1　グラフ表現の例

  - ノードのクラスタリング
  - ノードのランキング
  - ノードの特定
- エッジごとのタスク
  - リンク予測
- グラフ全体によるタスク
  - 部分グラフの特定
  - グラフ分類
  - グラフ生成モデルの作成

近年のグラフ構造を用いた研究の関心の高まりにより、これらのどのタスクにおいても近年多くの研究がされている。本研究ではこの中でもノード分類とリンク予測についてに注目し研究対象とした。

2.2 半教師ありノード分類

ノード分類はグラフ構造や特徴量からノードのラベルを予測するタスクである。その中でも半教師ありノード分類は、グラフ全体の構造、それぞれのノードにおける特徴量、少数のノードのラベルを用い、ラベルが与えられていない残りのノードのラベル予測をするタスクである。

グラフを $\mathcal{G} = (\mathcal{V}, \mathcal{E})$、ノード数を $n = |\mathcal{V}|$ とすると隣接行列は $\mathbf{A} \in R^{n \times n}$ となる。特徴量行列の集合を $\mathbf{X} = \{\boldsymbol{x_1}, ..., \boldsymbol{x_n}\}$、用いるラベルの個数を l とし、ノードの集合を $\mathcal{V} = \{v_1, ..., v_l, v_{l+1}, ..., v_n\}$ とする。またそれぞれのノードにおけるラベルをそれぞれ $y_1, ..., y_n$ とする。これらにより、半教師あり分類問題は以下のように定義される。

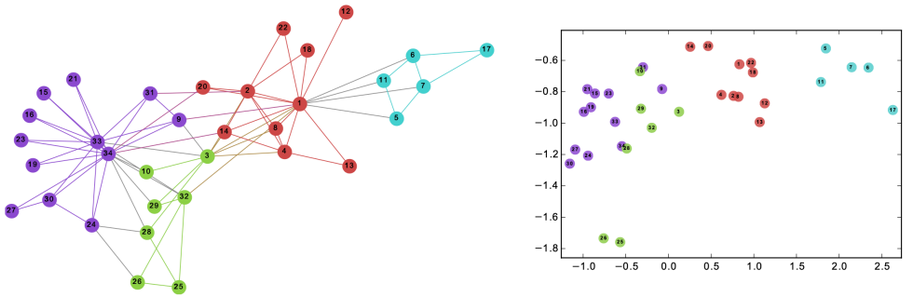
- 入力：
  隣接行列：$\mathbf{A} \in R^{n \times n}$,
  全ノードの特徴ベクトル：$\mathbf{X} = \{\boldsymbol{x_1}, ..., \boldsymbol{x_n}\}$,
  l 個の教師ありラベル： $y_1, ..., y_l$
- 出力：
  $n - l$ 個の教師なしラベル： $y_{l+1}, ..., y_n$

半教師あり学習のタスクににおける重要な特徴の 1 つは、学習においてクラスラベルは一部の教師データのみを扱うが、グラフ構造と特徴量は全てのノードの情報を用いることができるということである。それにより、ラベルがとても少ない場合においては特徴量とグラフ構造をいかにして学習に用いるかが重要となる。

半教師ありノード分類のタスクへの解法としては主にグラフベース学習、グラフエンベディング手法、グラフニューラルネットワークの 3 つが挙げられる。本研究で用いるグラフニューラルネットワークについては後の節で述べるとして、この節では簡単に前述した 2 つの手法について簡単に述べる。

グラフベース正則化

グラフベース正則化は古くから研究されていた手法である [4, 5]。グラフベース正則化の例としては Label Propagation [57] や Label Spreading [55] などがある。これらは「グラフ上で隣接しているノードペアやグラフ上で近くに存在するノードペアは等しいラベルをもつ」という多くのグラフ構造をもつデータセットにおける経験的な推論から予測する手法である。この手法は単純なモデルである一方、表現力が低いことやその推論が当てはまらないグラフデータにおける予測が難しいという問題がある。



(a) Input: Karate Graph　　(b) Output: Representation

図 2.2　グラフエンベディングの可視化の例 [35]

グラフエンベディング手法

エンベディング手法とはグラフ構造を用いることで、ノード, エッジ, グラフなどをベクトル空間に埋め込み、それぞれのノードを k 次元のベクトル表現で表す手法である [8]。図 2.2 はノードエンベディングの例を表している。例では左図の空手クラブのネットワークのノード 1 つ 1 つをベクトル空間に埋め込み、右図で 2 次元空間のベクトル表現で可視化している。一般的なノード分類の場合、この例のように同じ属性のノード同士をベクトル空間上で近くに埋め込むことで高い精度が得られることが多い。この手法は skip-gram [32] に基づいてグラフ上でのランダムウォークから分散表現を得るように学習する DeepWalk [35] と呼ばれる手法が提案されて以降、それを応用した様々な手法が提案されている [17, 36, 47, 49]。

2.3　リンク予測

リンク予測は与えられたグラフ構造から、情報が与えられていないノード間のエッジの有無を予測するタスクである。つまり一部のエッジの情報を $\mathcal{E}_{train} \subset \mathcal{E}$ とすると、入力としてグラフデータ $\mathcal{G} = (\mathcal{V}, \mathcal{E}_{train}, \mathbf{X})$ を用いて未知のエッジ $\mathcal{E} \setminus \mathcal{E}_{train}$ を予測するタスクである。また実世界上において、リンク予測は商品レコメンダシステム [56] や SNS での将来のフォロー関係の予測などの様々な分野に応用されている。

2.4　グラフ畳み込みネットワーク

2.4.1　グラフニューラルネットワーク

グラフ構造を含むデータに対するタスクを解く方法としてグラフニューラルネットワーク (Graph Neural Network) [38] が近年注目を集めている。GNN はグラフ構造に対して直接ニューラルネットワークを用いて解析する手法である [50]。この手法の強みとしてノードの特徴量だけでなく、エッジにより結びついた周囲のノードの情報を学習に用いることができることである。

GNN のフレームワークの 1 つに Neural Message Passing [15, 52, 54] が挙げられ、以下の手順で特徴量が更新される。

- AGGREGATE: 隣接ノードの情報の集約
  $a_v^{(k)} = AGGREGATE^{(k)}(\{h_u^{(k-1)} : u \in \mathcal{N}(v)\})$
- COMBINE: 隣接行列に基づいてノードの特徴量を結合
  $h_v^{(k)} = COMBINE^{(k)}(h_v^{(k-1)}, a_v^k)$

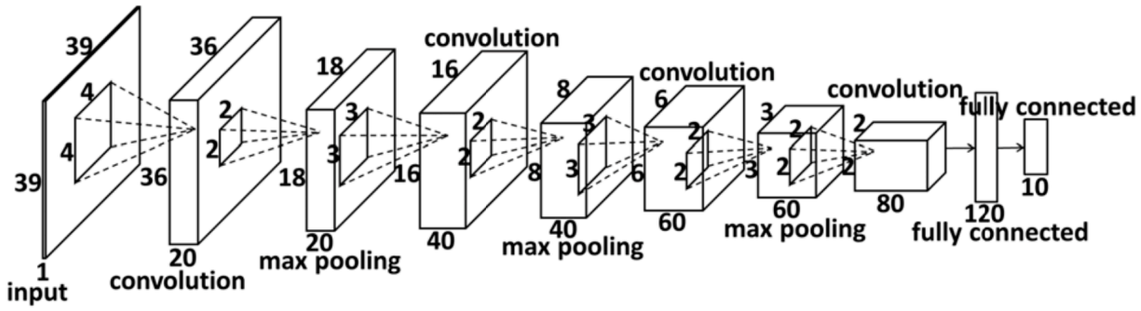


図 2.3　畳み込みニューラルネットワークの例 [45]

- READOUT: 得られた特徴量の呼び出し
  $h_G = READOUT(h_v^k | v \in \mathcal{G})$

これらの演算は入力として無向グラフ $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ とノード $v \in \mathcal{G}$ それぞれに特徴量 h_v が与えられた際に、ノード v に対して GNN を適用した式である。AGGREGATE でノード v の近隣ノードの集合 $\mathcal{N}(v)$ から特徴量を集約し、COMBINE で自身ノードと近隣ノードの特徴量を結合し新しい特徴量を求め、READOUT でグラフ全体の特徴量を求めている。Neural Message Passing の AGGREGATE や COMBINE 関数を具体的に定義することにより GCN [24],GAT [48],MoNet [34],GN [3] といった様々なモデルを作成することができる [51]。

2.4.2　畳み込みニューラルネットワーク

ディープラーニングの研究が盛んに行われるようになった大きな要因として畳み込みニューラルネットワーク (Convolutional Neural Network) を用いた手法 [26] が Image Net Large Scale Visual Recognition Challenge(IRSVRC) において他の手法に圧倒的な差をつけて優勝したことが挙げられる。畳み込みニューラルネットワークのモデルは、畳み込み演算をするフィルタを移動させることで、位置普遍性や移動普遍性を抽出し学習することができる。また畳み込み演算において、重み変換や行列の次数削減などにより汎化性能を高めることができる。

しかしグラフ構造を持つデータは画像データとは異なり、各ノードの接続関係が不規則であるため同様の畳み込み演算を用いることはできない。そこでグラフデータに対して畳み込み演算を定義し試みる手法としてグラフフーリエ変換を用いる spectral convolution と直接的に行う spatial convolution の 2 種類の手法が提案された。

2.4.3　spectral convolution

spectral convolution [6] はグラフにおける信号処理の考え方 [40] に基づいてグラフ上で畳み込み演算を定義する手法である。この手法はまずグラフ上のデータをグラフラプラシアンの固有値分解を用いて信号の傾きにより成分分解し、求められた要素積を逆フーリエ変換によりノードの情報を更新する手法である。図 2.4 ではこの手法をグラフに適用した例を示している。

しかしこの手法には

- 固有値分解の計算量が大きい
- グラフ上では空間的に局在化されたフィルターを得ることができない
- セルフループや多重エッジに適用できない

などの問題がある。

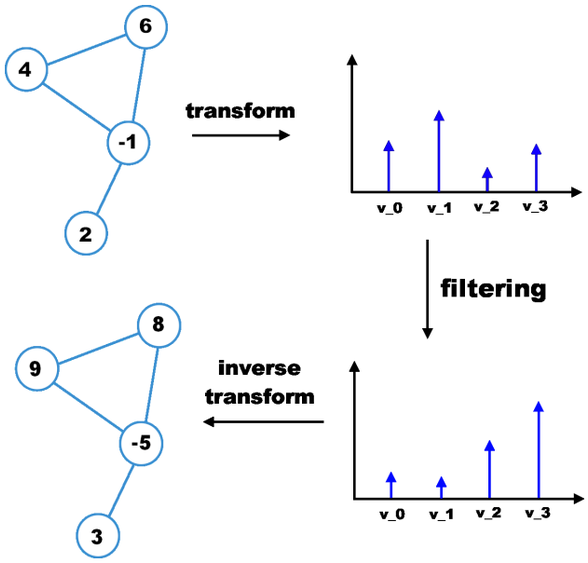


図 2.4　spectral convolution [6]

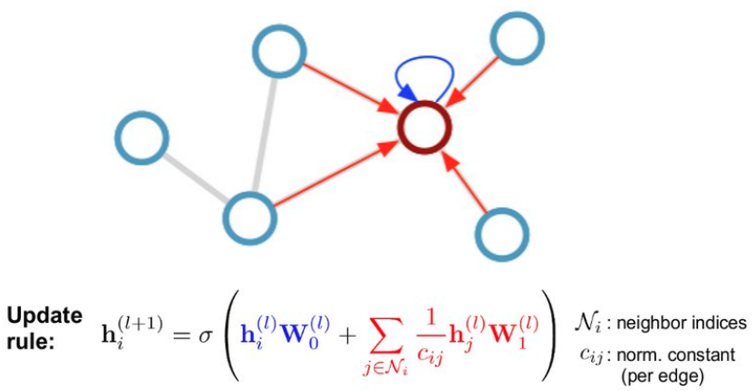


図 2.5　spatial convolution [24]

2.4.4　spatial convolution

spatial convolution [16] は spectral convolution の問題点である計算量が大きいことやセルフループや多重辺を持つグラフに適用できない問題点を解決した手法である。この手法ではグラフ構造の近隣ノードの信号の情報を用いて自身のノードの情報を更新していくことにより畳み込みを定義する。ノード i における畳み込み演算は以下の式で表せる。

$$h_i^{(l+1)} = \sigma\left(h_i^{(l)}W_0^{(l)} + \sum_{j\in N_i}\frac{1}{c_{i,j}}h_j^{(l)}W_1^{(l)}\right). \tag{2.1}$$

ここで N_i はノード i にエッジを張るノードの集合であり、$c_{i,j}$ は正規化のための定数である。また $W_0^{(l)}$、$W_1^{(l)}$ は第 l 層における隠れ層の重み行列である。

2.5 グラフ畳み込みネットワーク

グラフ畳み込みネットワーク (Graph Convolutional Network;GCN) [24] は spactral convolution を spatial convolution としても解釈できることを示した手法である。グラフ畳み込みネットワークの第 i 層の出力は以下の式で表せる。

$$H^{(i)} = \sigma(D^{-1/2}\tilde{A}D^{-1/2}H^{(i-1)}W^{(i)}) \tag{2.2}$$

ここで、$I_n \in \mathcal{R}^{n\times n}$ は単位行列、$\tilde{A} = A + I_n$、$D \in \mathcal{R}^{n\times n}$ は次数行列、$H^{(0)} = X$、$W^{(i)}$ は i 層における特徴行列を更新するための重み行列である。また、σ は ReLU 関数や sigmoid 関数などの活性化関数である。

GCN と同様にグラフ上で畳み込み演算をする手法として、エッジに Attention をかけて学習する GAT [48] や、Mean Aggregator や LSTM Aggregator などを用いた GraphSAGE [19]、weisfeiler-lehman テストを用いた GIN [51]、Mix-hop-propagation を用いる MixHop [1] といった手法が後に提案された。

GCN は層を重ねるごとにノードの情報量が指数関数的に喪失する oversmoothing が発生するため一般的には 2 層 GCN が用いられるが [27]、oversmoothing を回避するための手法 [29, 33] が近年提案された。またノード数が数億を超える大きいグラフに対応させるために GCN の計算量を削減するためにサンプリングを用いた手法 [9, 19] やバッチ学習を行う手法 [10] も提案されている。

2.5.1 半教師ありノード分類

GCN を用いて半教師ありノード分類するタスクにおいては、一般的にグラフ $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ を表す隣接行列 $\mathbf{A}$ とノード特徴量 $\mathbf{X}$、一部ノードのラベル $\boldsymbol{y}_{train} \in \mathcal{Y}$ を入力データとして、ラベルの与えられていない残りのノードのラベル $\mathcal{Y}$ を予測する精度を測る。

GCN は一般的には 2 層用いられ、2 層 GCN は以下の式で表せる。

$$\hat{\mathbf{Y}} = \mathrm{GCN}(\mathbf{X}, \mathbf{A}) \coloneqq \mathrm{softmax}(\hat{\mathbf{A}}\mathrm{ReLU}(\hat{\mathbf{A}}\mathbf{X}\mathbf{W}^{(1)})\mathbf{W}^{(2)}) \tag{2.3}$$

ここで $\hat{\mathbf{Y}} = [\hat{\boldsymbol{y}}_1, \ldots, \hat{\boldsymbol{y}}_N]^\mathsf{T} \in [0,1]^{N\times C}$ はノードのラベル予測を行列で表している。また GCN はノードごとに $\hat{\boldsymbol{y}}_i = [\hat{y}_{i1}, \ldots, \hat{y}_{iC}]^\mathsf{T}$ の出力によりそれぞれのノードのラベルの予測を行列で示している。ここで $0 \leq \hat{y}_{ic} \leq 1$ かつ $\sum_c \hat{y}_{ic} = 1$ という制約の中で $\hat{y}_{ic}$ は、ノード i がクラス c に属する確率を表す。

他クラス分類においては、交差エントロピー誤差を用いることで学習の損失関数は以下のように計算できる。

$$L_{supervised} = -\sum_{i\in\mathcal{V}_{train}}\sum_{c=1}^{C} y_{ic}\log\hat{y}_{ic} \tag{2.4}$$

ここで $\boldsymbol{y}_i \in \{0,1\}^C$ はラベル付けされたノードの場合 1、それ以外で 0 で表されるような one-hot ベクトルを用いて各ノードの正解ラベルを表す。

2.5.2 リンク予測

リンク予測は一般的にはグラフ $\mathcal{G} = (\mathcal{V}, \mathcal{E}_{train})$ を表す隣接行列 $\mathbf{A}_{train}$、ノード特徴量 $\mathbf{X}$ を用いて、情報が与えられていない 2 ノード間のエッジの有無を予測するタスクである。ここで $\mathcal{E}_{train}$ は完全データからエッジの教師データのみを抽出した集合である。

GCN を用いてリンク予測を行うモデルとして提案手法で用いる VGAE [23] について述べる。VGAE は変分オートエンコーダ (Variational Auto Encoder;VAE) [22] をグラフに適用したモデルである。

VGAE はエンコーダとデコーダから構成される。エンコーダでは GCN を用いてノードの特徴量と隣接行列から確率的な潜在表現を学習し、デコーダでは内積とシグモイド関数を用いて元の隣接行列を復元する。

まずエンコーダについて述べる。本章では 2 層エンコーダを用いた計算について記す。まず確率変数 $\boldsymbol{z}_i \in \mathbb{R}^F$ を用いて平均 $\boldsymbol{\mu}_i \in \mathbb{R}^F$ と対角共分散行列 $\mathrm{diag}(\boldsymbol{\sigma}_i) \in \mathbb{R}^{F \times F}$ は以下の式で計算できる。

$$[\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_N] = \mathrm{GCN}_{\boldsymbol{\mu}}(\mathbf{X}, \mathbf{A}) \coloneqq \hat{\mathbf{A}}\mathrm{ReLU}(\hat{\mathbf{A}}\mathbf{X}\mathbf{W}^{(1)})\mathbf{W}^{(2)}_{\boldsymbol{\mu}} \tag{2.5}$$

$$[\boldsymbol{\sigma}_1, \ldots, \boldsymbol{\sigma}_N] = \mathrm{GCN}_{\boldsymbol{\sigma}}(\mathbf{X}, \mathbf{A}) \coloneqq \hat{\mathbf{A}}\mathrm{ReLU}(\hat{\mathbf{A}}\mathbf{X}\mathbf{W}^{(1)})\mathbf{W}^{(2)}_{\boldsymbol{\sigma}} \tag{2.6}$$

ここで $\mathrm{GCN}_{\boldsymbol{\mu}}$ と $\mathrm{GCN}_{\boldsymbol{\sigma}}$ の計算において GCN 1 層目に用いる重みパラメータ $\mathbf{W}^{(1)}$ を共有している。また確率変数 $\boldsymbol{z}_i \in \mathbb{R}^F$ と求められた平均平均 $\boldsymbol{\mu}_i \in \mathbb{R}^F$ と対角共分散行列 $\mathrm{diag}(\boldsymbol{\sigma}_i) \in \mathbb{R}^{F \times F}$ を用いるとエンコーダは以下の式で計算できる。

$$q(\mathbf{Z}|\mathbf{X}, \mathbf{A}) = \prod_{i=1}^{N} q(\boldsymbol{z}_i|\mathbf{X}, \mathbf{A}) \tag{2.7}$$

$$= \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{z}_i|\boldsymbol{\mu}_i, \mathrm{diag}(\boldsymbol{\sigma}_i^2)) \tag{2.8}$$

次にデコーダについて述べる。デコーダはノードペアの潜在変数の内積により隣接行列を復元する。

$$p(\mathbf{A}|\mathbf{Z}) = \prod_{i=1}^{N}\prod_{j=1}^{N} p(A_{ij}|\boldsymbol{z}_i, \boldsymbol{z}_j) \tag{2.9}$$

ここで $p(A_{ij} = 1|\boldsymbol{z}_i, \boldsymbol{z}_j) = \mathrm{sigmoid}(\boldsymbol{z}_i^{\mathsf{T}}\boldsymbol{z}_j)$ とすることで $\boldsymbol{z}_i$ と $\boldsymbol{z}_j$ の内積がエッジの存在確率を表すようにしている。

最後に損失関数について述べる。潜在変数の事前分布を $p(\mathbf{Z}) = \prod_i p(\boldsymbol{z}_i) = \mathcal{N}(\boldsymbol{z}_i|\mathbf{0}, \mathbf{I})$、潜在変数を $\boldsymbol{z}_i = \boldsymbol{\mu}_i + \boldsymbol{\epsilon} \odot \boldsymbol{\sigma}_i$(ノイズを $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$、要素積を $\odot$ として表す) とすると損失関数は以下の式で表せる。

$$\mathcal{L} = \mathbb{E}_{q(\mathbf{Z}|\mathbf{X},\mathbf{A})}[\log p(\mathbf{A}|\mathbf{Z})] - \mathrm{KL}[q(\mathbf{Z}|\mathbf{X}, \mathbf{A})||p(\mathbf{Z})] \tag{2.10}$$

隣接行列 $p(\mathbf{A})$ を直接求めるためには計算量的に困難であるため、変分下限を最大化する目的関数を用いている。

2.6 欠損データ

2.6.1 実世界上の欠損値

実世界において欠損値を含むデータは数多く存在する。

- 人的なミス
- センサーによるエラー入力
- 任意回答における未入力項目を含むアンケート
- ビッグデータの情報欠損

このような欠損値を含むデータ構造は図 2.6 のようなデータとして保持される。

ID	給料	住所	年齢
1	500万	東京都	30
2	400万	?	45
3	?	北海道	28

図 2.6　欠損値の例 (ユーザー情報)

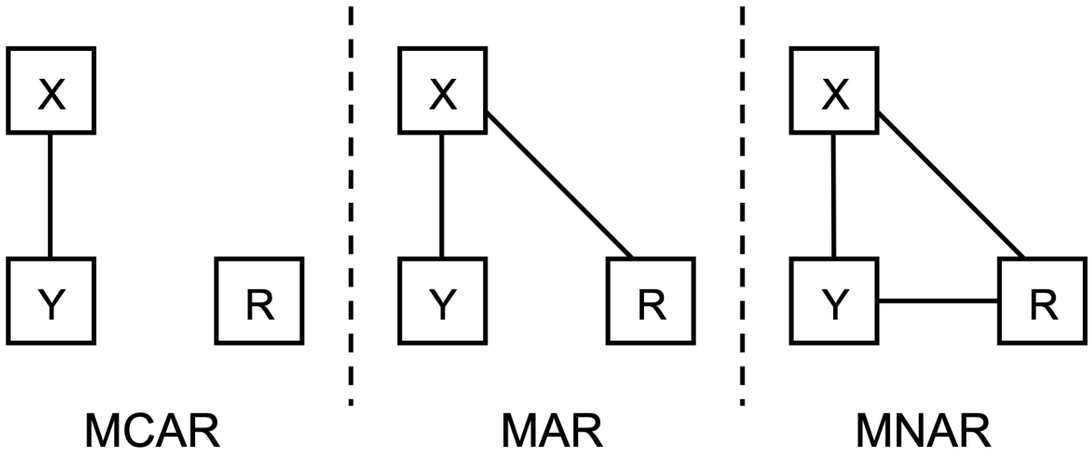


図 2.7　欠損パターンの概念図。R を欠損識別変数、X を観測値データ、Y を欠損値データとして表している。

2.6.2　欠損値のパターンと表現

データ中の一部の値が不明のデータは一般に欠損データ（missing data）と呼ばれる。欠損データは値が分かる観測値（observed value）と値が不明の欠損値（missing value）から構成される。欠損値を一つも含まないデータは完全データ（complete data）と呼ばれ、欠損値を 1 つでも含むデータを不完全データ（incomplete data）と呼ばれている。

R を欠損識別変数、X_o を観測値データ、X_m を欠損値データとすると、ある特徴量が欠損している確率は条件付き確率を用いて $p(\mathbf{R}|\mathbf{X}_o, \mathbf{X}_m)$ と表すことができる。また、Little らによると欠損値のパターンは 3 つに分類することができる [28]。

- MCAR (Missing Completely At Random):
  MCAR はデータ構造に関係なく、欠損値となるデータが完全にランダムに決定されたケースである。条件付確率は $p(\mathbf{R}|\mathbf{X}_o, \mathbf{X}_m) = p(\mathbf{R})$ と表現できる。図 2.7 の左図は MCAR の概念図である。
- MAR (Missing At Random):
  MAR はデータの欠損している確率がそのデータの種類により決定されたケースである。そのため MAR は MCAR よりも制約が弱く、MAR は MCAR をより抽象化したケースと言える。条件付確率は $p(\mathbf{R}|\mathbf{X}_o, \mathbf{X}_m) = p(\mathbf{R}|\mathbf{X}_{obs})$ と表現できる。図 2.7 の中央図は MAR の概念図である。
- MNAR (Missing Not At Random):
  MNAR は欠損値を持つ変数がそれぞれの欠損値の有無により決定されるケースである。条件付確率は $p(\mathbf{R}|\mathbf{X}_o, \mathbf{X}_m) \neq p(\mathbf{R}|\mathbf{X}_{obs})$ と表現できる。図 2.7 の右図は MNAR の概念図である。

これらの欠損値のパターンでは MCAR、MAR、MNAR の順番で扱いやすい一方、現実世界の複雑

Col1	Col2	Col3
1	b	T
2	NA	T
3	b	T
4	b	NA
5	a	F

(a)不完全データ

Col1	Col2	Col3
1	b	T
3	b	T
5	a	F

(b)リストワイズ除去

Col1	Col2	Col3
1	b	T
3	b	T
4	b	NA
5	a	F

(c)ペアワイズ除去

図 2.8　欠損値の除去の例

性から、欠損値を含む実データの多くは MNAR であることが知られている。

2.7　欠損値補完

欠損値を扱う方法は以下の 3 つに分類することができる。

- 欠損値の除去
- 代入法（imputation）
- 不完全データとして扱う

2.7.1　欠損値の除去

欠損値の除去は欠損値を含む項目をデータセットから取り除く手法である。欠損値を除去する方法はリストワイズ除去とペアワイズ除去の 2 種類がある。図 2.8 は不完全データ (a) に対する欠損値除去の例を (b)、(c) で示している。(b) のリストワイズ除去 [13] は解析対象の持つ変数のうちどれか一つでも欠損値を持つケースのデータを削除する方法である。この手法は欠損率が高い場合に情報量がとても小さくなってしまう問題がある。(c) のペアワイズ除去は変数のうちどれか一つでも欠損値を持つケースのデータを選択的に削除する方法である。リストワイズ除去と比べて情報を多く保持できる一方、欠損値を含むためデータが扱いづらい問題がある。

2.7.2　代入法 (imputation)

代入法とは欠損値を何らかの方法を用いて補完することで不完全データを完全データにする方法である。図 2.9 は代入法を用いて不完全データ (a) を完全データ (b) に変換している。ここで X_{ij} は定数である。代入法は完全データが得られるため、得られたデータに既存の機械学習手法を適用することができるという強みがある一方、誤った欠損値補完をすることで精度を下げてしまう可能性があるという問題がある。

2.7.3　不完全データとして扱う

欠損値をもつ不完全データをそのまま扱う手法がある。この場合は観測値の情報を欠損なしで扱うことができ、また代入法とは異なり誤ったデータを用いることなく解析することができるが、データに欠損値を含むためデータの扱いが難しいという問題がある。GMMC [42] を用いて学習する手法 [46] やアンサンブル学習を用いた手法 [20]、オートエンコーダを用いた手法 [41] などがある。

Col1	Col2	Col3
1	b	T
NA	NA	NA
3	b	T
4	b	NA
5	a	F

(a)不完全データ

Col1	Col2	Col3
1	b	T
X_{21}	X_{22}	X_{23}
3	b	T
4	b	X_{43}
5	a	F

(b)完全データ

図 2.9　代入法 (imputation) の例

第 3 章

提案手法

本章では GCN を欠損値データに適用するための欠損値補完を用いた 2 つの手法を提案する。

3.1 構造的補完

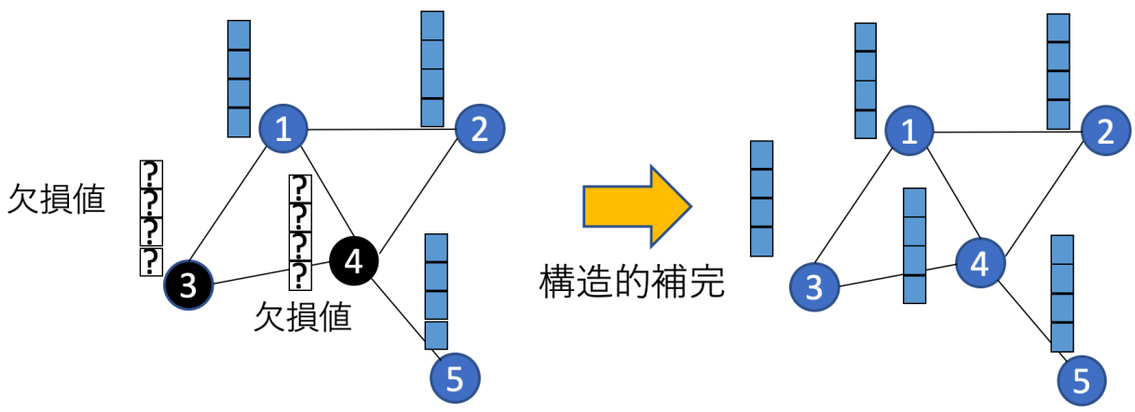


図 3.1 構造的補完

欠損値を代入法を用いて補完する手法のうち、グラフ構造を用いて欠損値補完をする手法を構造的補完と定義する。図 3.1 は構造的補完の例を示している。この図では便宜的にノードの全ての特徴量が欠損しているノードと特徴量の全てが欠損していないノードの 2 種類のノードから成り立つグラフを用いている。図 3.1 の左図はノード番号 3 と 4 のノードの特徴量が全て欠損しており、ノード番号 1,2,5 のノードの特徴量は欠損していないケースを示している。図 3.1 の左図に構造的補完を適用することで右図のように全てのノードに欠損値のないグラフを得ることができる。構造的補完により求められる欠損値を補完された特徴量 X'_{ij} は以下の式 (3.1) のように定義できる。

$$X'_{ij} = \begin{cases} X_{ij} & (r_{ij} = 1) \\ F(X_{ij}, A) & (r_{ij} = 0) \end{cases} \tag{3.1}$$

ここで X_{ij} は欠損値を含む特徴量、A は隣接行列、F は構造的補完の方法により決まる関数である。r_{ij} は x_{ij} が観測値の場合には 1 で x_{ij} が欠損値の場合には 0 と定義する。

3.2 近隣平均補完

本節では提案手法で用いる近隣平均補完について述べる。近隣平均補完は構造的補完の 1 つであり、特徴量の種類ごとに欠損値補完を行う手法である。欠損しているノードの特徴量を、そのノードの近隣

ノードのうち欠損していないノードの特徴量の平均を用いて補完する方法を近隣平均補完と定義する。またもしそのノードの近隣ノードの全てが欠損値を含んでいる場合は 0 を特徴量として代入する。近隣平均補完により求められる欠損値を補完された特徴量 X'_{ij} は以下の式 (3.2) のように定義できる。

$$X'_{ij} = \begin{cases} X_{ij} & (r_{ij} = 1) \\ \frac{1}{|N'_i|} \sum_{k \in N'_i} X_{kj} & (r_{ij} = 0 \quad \& \quad |N'_i| \neq 0) \\ 0 & (r_{ij} = 0 \quad \& \quad |N'_i| = 0) \end{cases} \tag{3.2}$$

ここで X_{ij} は欠損値を含む特徴量、N'_i はノード i の近隣ノードの中で欠損値を含まないノードの集合であり、r_{ij} は x_{ij} が観測値の場合には 1 で x_{ij} が欠損値の場合には 0 と定義する。

図 3.2 は近隣平均補完の例を示している。左図ではノード番号 3 と 4 のノードの特徴量が全て欠損しており、ノード番号 1,2,5 のノードの特徴量は欠損していないケースである。近隣平均補完を適用することでノード 3 とノード 4 の欠損値はそのノードの近隣ノードのうち欠損していないノードの特徴量の平均を用いて補完できる。つまり、ノード 3 の特徴量はノード 1 の特徴量と等しい値で補完され、ノード 4 の特徴量はノード 1,2,5 の特徴量の平均値で補完される。

近隣平均補完はグラフ構造を用いて補完することができる一方、欠損ノードの近隣ノードの多くが欠損している場合や近隣ノードの特徴量が全て欠損している場合に補完の精度が低くなる問題がある。次節では近隣平均補完の問題点を改善した近隣再帰補完の詳細を述べる。

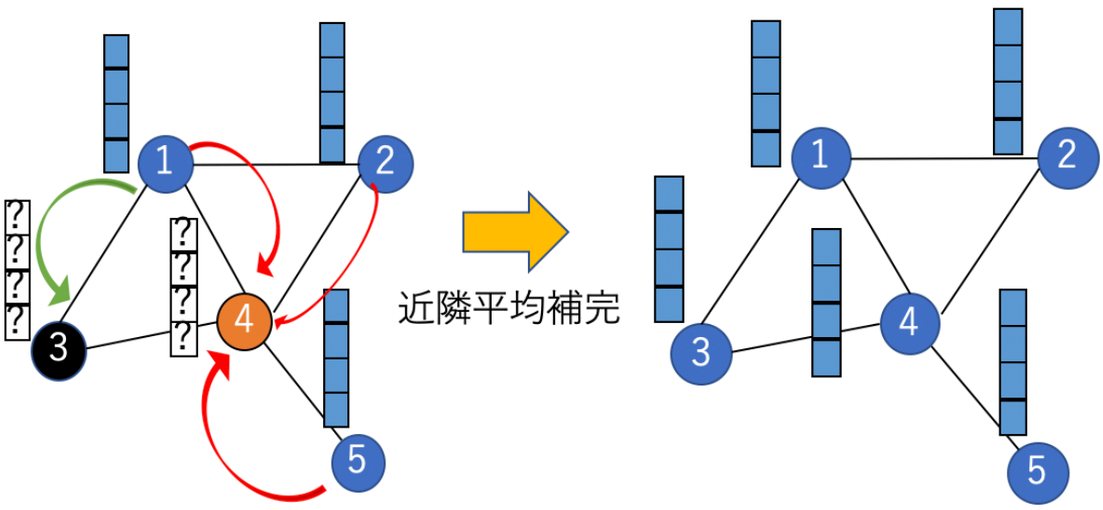


図 3.2　近隣平均補完

3.3　近隣再帰補完

本節では提案手法で用いる近隣再帰補完について述べる。近隣再帰補完は構造的補完の 1 つであり、特徴量の種類ごとに再帰的に補完する手法である。近隣再帰補完は以下の 2 ステップから成り立っている。

(1) 欠損している全ての特徴量を 0 で埋める
(2) 元が欠損値であった特徴量だけをそのノードと近隣ノードの特徴量の平均で更新する

近隣再帰補完は (1) で欠損値を 0 埋めした後に (2) で欠損値の補完を再帰的に決められた回数繰り返す手法である。(2) の特徴量の更新 1 回で求まる特徴量 $X^{(l)}_{ij}$ は以下の式 (3.3) のように定義できる。

$$X_{ij}^{(l)} = \begin{cases} X_{ij}^{(l-1)} & (r_{ij} = 1) \\ \frac{1}{|N_i|} \sum_{k \in N_i} X_{kj}^{(l-1)} & (r_{ij} = 0) \end{cases} \tag{3.3}$$

ここで $X_{ij}^{(l)}$ は l 回の再帰後に得られた特徴量であり、$X_{ij}^{(0)} = X_{ij}$ と定義する。N_i はノード i の近隣ノードの集合であり、r_{ij} は x_{ij} が観測値の場合には 1 で x_{ij} が欠損値の場合には 0 と定義する。

図 3.3 は近隣再帰補完の例を示している。左図ではノード番号 3 と 4 のノードの特徴量が全て欠損しており、ノード番号 1,2,5 のノードの特徴量は欠損していないケースである。近隣再帰補完を適用することでノード 3 とノード 4 の欠損値はまずステップ (1) に従って 0 で補完される。その後ステップ (2) に従って、元が欠損値であったノードの特徴量は近隣ノードとそのノードの特徴量の平均、つまりノード 3 はノード 1,3,4 の特徴量の平均で補完され、ノード 4 はノード 1,2,3,4,5 の特徴量の平均で補完される。この (2) の操作を決められた回数行い得られた特徴量を用いてノード 3 とノード 4 の欠損値を補完する。

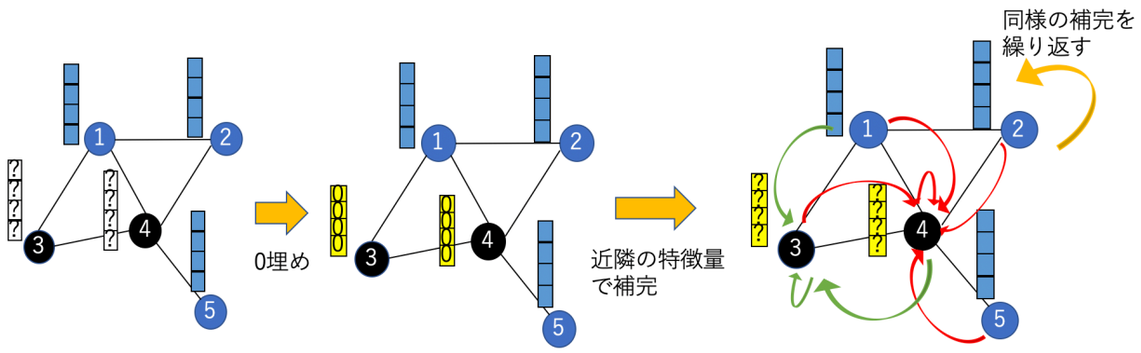


図 3.3　近隣再帰補完

3.4　GCN

提案手法で扱う 2 層のグラフ畳み込みネットワークについて述べる。1 層目の畳み込みネットワークを GCN_1、2 層目の畳み込みネットワークを GCN_2 とすると以下の式 (3.4) で表せる。

$$H_{hid} = GCN_1(X^{'}, A) \tag{3.4}$$
$$H_{out} = GCN_2(H_{hid}, A) \tag{3.5}$$

ここで、$X^{'}$ は欠損値補完により得られた特徴量、A は隣接行列、H_{hid} は GCN 1 層目での出力、H_{out} は GCN2 層目での出力である。

3.5　提案手法の定義

本研究では構造的補完とグラフ畳み込みネットワークを組み合わせたモデルの 2 つを定義し提案する。提案手法の全体的な流れは図 3.4 で示している。1 つ目の提案手法 GCN_neighbor モデルはまず近隣平均補完で完全データを得た後、MLP を用いて予測するモデルである。2 つ目の提案手法 GCN_recursive モデルはまず近隣再帰補完で完全データを得た後、MLP を用いて予測するモデルである。ここでどちらの提案手法においてもノード分類のタスクにおいては MLP として 2 層 GCN を用い、リンク予測のタスクにおいては VGAE を用いた。ここで VGAE はエンコーダを 2 層 GCN、デコーダを内積とした。

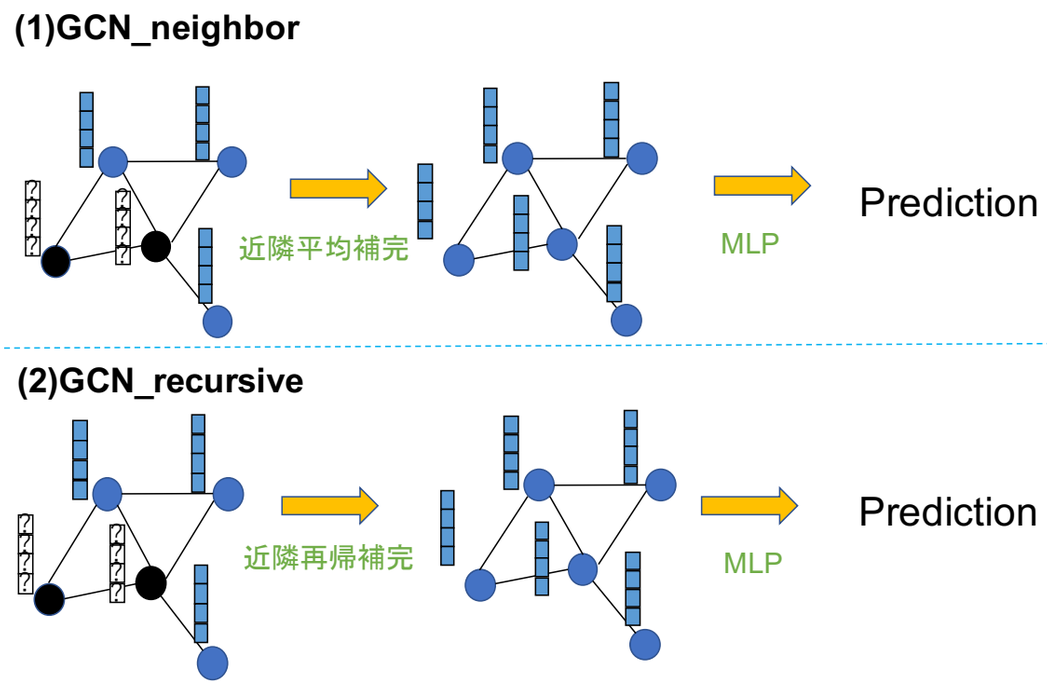

(1)GCN_neighbor
近隣平均補完
MLP
Prediction
(2)GCN_recursive
近隣再帰補完
MLP
Prediction

図 3.4　提案手法の概要

第 4 章

実験

4.1 実験の設定

4.1.1 提案手法の概要

提案手法のモデルについて述べる。実装は Python を使用言語とし、PyTorch をフレームワークとして用いた。提案手法には 2 層のグラフ畳み込みネットワークを用いた。畳み込み層では過学習を防ぐための dropout を用い、活性化関数として、隠れ層では ReLU 関数、出力層として Softmax 関数を用いた。また最適化には Adam [21] を用いた。また提案手法や比較手法で用いる GCN のハイパーパラメータは表 4.1 に示した通り、ノード分類とリンク予測では異なる値を用いた。

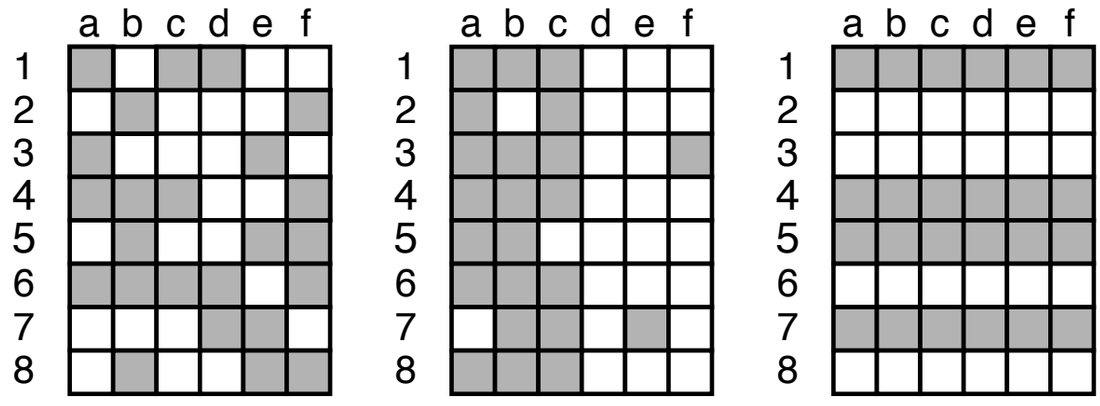
4.1.2 データセット

実験には引用ネットワーク (Cora,Citeseer) [39] と共購買ネットワーク (Amaphoto,Amacomp) [31] の計 4 種類のデータセットを用いた。

- 引用ネットワーク (Cora, Citeseer):
  論文をノード、引用をエッジとしたネットワークである。論文を bag-of-words 表現で表し、その単語が含まれる場合を 1、それ以外を 0 としてベクトル表現で表したものを特徴量としている。論文の種類をラベルとして各ノードに割り当てている。
- 共購買ネットワーク (Amaphoto, Amacomp):
  商品をノード、利用者が同時に購入しやすい商品をエッジで表したネットワークである。その商品に対するユーザーのレビューを bag-of-words 表現で表したものを特徴量としている。商品の種類をラベルとして各ノードに割り当てている。

4 種類のデータセットの詳細は以下の表 4.2 で示した。データセットは比較手法の条件と同様になるように、有向エッジを全て無向エッジに変換して実験を行った。また実験を行うためにデータセットに人工的に欠損値を含むような実装をした。人工的な欠損値として以下の 3 種類の欠損パターンを用いた。

- Unirom randomly missing:
  全ての特徴量を等しい確率で欠損させたケースである。
- Biased randomly missing:
  特徴量の種類ごとにランダムに特徴量を選択し、選択された特徴量の欠損率を 90% に、選択されなかった特徴量の欠損率を 10% にしたケースである。
- Structurally missing:



(a) Uniform randomly missing (b) Biased randomly missing (c) Structurally missing

図 4.1　欠損率 50% のときの欠損パターン例。8 ノード（ノード番号 1 – 8）、6 次元（a – f）。白いセルは観測されている要素を、黒いセルは欠損している要素を表す。[46] より引用。

ノード全体の集合から一様ランダムにノードを選択し、選択されたノードの特徴量を全て欠損させるケースである。

図 4.1 は特徴量にそれぞれの欠損パターンを適用させた例を示している。この例は欠損率を 50% にした例である。またグラフにおいては縦軸をノード番号、横軸を特徴量番号として考えることができる。(a) の Uniform randomly missing は全ての特徴量の中からランダムで 50% の特徴量が欠損している。(b) の Biased randomly missing では特徴量 a,b,c では 90% のノードを欠損値としており、d,e,f では 10% のノードを欠損値としている。(c) の Structurally missing ではノード 1,4,5,7 の全ての特徴量を欠損値であり、それ以外の全ての特徴量を観測値となっている。

4.1.3　比較手法

比較手法では代入法を用いた 9 つの手法と欠損データを直接学習する 1 つの手法を用いる。

- MEAN [13]: 特徴量の種類ごとに観測値のみで平均値を調べ、特徴量の種類ごとの欠損値にその値を代入する。
- K-NN [2]: K 近傍法を用いて、それぞれのノードごとの特徴量ベクトルに類似した特徴量ベクトル K 個の平均により欠損値を代入法により補完する。
- MFT [25]: Matrix Factorization を用いて全体の特徴量行列を 2 つの低ランク行列に分解し、再構成した値により欠損値を補完する。
- SoftImp [30]: 特徴量行列を特異値分解することで低ランクの近似を行い行列補完する。

表 4.1　GCN のハイパーパラメータ設定

	学習率	L2 正則化項	ドロップアウト率	エポック数	patience
ノード分類（Cora、Citeseer）	0.01	$5 \cdot 10^{-4}$	0.5	200	10
ノード分類（AmaPhoto、AmaComp）	0.005	$5 \cdot 10^{-5}$	0.5	10000	100
リンク予測	0.01	0	0	200	-

表 4.2　データセットの詳細情報。ただし、訓練ラベル数、検証ラベル数、テストラベル数はそれぞれ半教師ありノード分類で用いるラベルの数に対応する。[46] より引用

データセット	Cora	Citeseer	AmaPhoto	AmaComp
ノード数	2,708	3,327	7,650	13,752
エッジ数	5,429	4,732	143,663	287,209
特徴量の次元	1,433	3,703	745	767
クラス数	7	6	8	10
訓練ラベル数	140	120	320	400
検証ラベル数	500	500	500	500
テストラベル数	1,000	1,000	6,830	12,852

- MICE [7]: 多重代入法 [37] の 1 つである。連鎖方程式に基づいて得られた複数のデータを用いて欠損値を補完する。
- MissForest [44]: ランダムフォレストを用いて欠損値を補完する。
- VAE [22]: 変分オートエンコーダを用いる手法である。欠損値を潜在変数を用いて補完する。
- GAIN [53]: 敵対的生成ネットワークを用いて欠損値補完する。
- GINN [43]: Graph Denoising AutoEncode を用いて欠損値補完する。
- GCNmf [46]: 混合ガウスモデル (GMM) を用いて欠損値を確率分布で表現し、欠損値を含む特徴量を用いて直接学習する。

4.2　ノード分類

提案手法の評価を行うために半教師ありノード分類の精度を調べた。実験は Uniform randomly missing, Biased randomly missing, Structurally missing の 3 種類の欠損パターンを用いた。また、欠損率は 10% から 90% までを 10% づつ変化させた 9 種類の欠損率を用いた。ラベルの訓練/検証/テストの割合は Cora,Citeseer ではクラスごとに 20 ノードを訓練データとして選択し、検証ラベル数は 500, テストラベル数は 1000 で実験を行った。また Amaphoto,Amacomp ではクラスごとに 40 ノードを訓練データとして選択し、検証ラベル数は 500, 残りの全てのノードをテストラベルとしてで実験を行った。また実験は 1 つの欠損パターンで 20 回の実験を行い、それを 5 つの欠損パターンで行い平均することで精度を調べた。つまり合計で 100 回実験の平均を分類精度とした。

分類精度の指標はそのノードのラベルの正解率を Accuracy として求めた。Accuracy は 0 から 100 までの値をとり、値が大きいほど精度が良いことを示す。実験のエポック数を 100,000 とし、その中で Early Stopping の patience を 100 とすることで学習を打ち切りにすることで過学習を防いだ。また GCN を用いる手法において隠れ層の次元数は Cora,Citeseer で 16、Amaphoto,Amacomp で 64 に統一した。提案手法の 1 つである GCN_recursive モデルの再帰回数は 32 回として実験を行った。

表 4.3 から表 4.6 はノード分類の精度を示している。欠損パターンと欠損値率毎のそれぞれの結果で最も良い精度を示している値を太字として示している。比較手法の中で、MICE の Cora,Citeseer の結果と、MissForest の Citeseer,AmaComp は実験が 24 時間以内に正常終了しないため、値を空欄で示している。

また表の最下層の列で完全データの特徴量を用いて実験を行った場合の GCN の精度と、完全データの特徴量を用いる代わりに、隣接行列を単位行列として実験を行った場合の GCN の精度を GCN ($\mathbf{X} = \mathbf{I}_N$) として提示している。

表 4.3 から表 4.6 の結果から以下のような考察をすることができる。まずはデータセットごとの結果を考える。データセット Cora においては 3 つの欠損パターンにおいてほとんどの欠損率で提案手法の GCN_recursive モデルの精度が比較手法を上回っていることがわかる。データセット Citeseer,Amaphoto,Amacomp においては、3 つの欠損パターンにおいてほとんどの欠損率において提案手法である GCN_recursive モデルと比較手法の 1 つである GCN_mf モデルが高い精度を示している。

実験の多くの欠損パターン/欠損率で提案手法が比較手法よりも高い精度を示した要因を考察する。代入法を用いた比較手法は欠損値補完にグラフ構造を用いていない。また欠損値を含む特徴量から直接学習を行う手法である GCN_mf モデルは GMM での特徴量表現においてノード間の関係を考慮していない。一方、本研究の 2 つの提案手法はグラフ構造を用いて欠損値補完を行っている。そのため、GCN がノードの特徴量を近隣ノードの特徴量を用いて学習するようなグラフを用いるモデルであることを考慮すると、提案手法は比較手法に比べてより GCN に適した欠損値補完をしていると考えられる。

また Cora データセットで特に提案手法が高い精度を示している要因を考察する。Cora データセットは Amaphoto や Amacomp と比べて疎なグラフであり、特徴量の種類も Citeseer に比べると少なくなっている。そのため、特徴量 1 つ 1 つの情報が他のデータセットに比べると重要であると考えられる。そのためグラフ構造を用いて GCN に適した欠損値補完をすることが、他のデータセットと比べてより効果を発揮していると考えられる。

次に欠損率が高い場合の提案手法と比較手法の精度を比較し考察する。欠損率が 90% の場合の実験結果から考える。全てのデータセット、全ての欠損パターンにおいて、欠損率が 90% の場合は提案手法が比較手法よりも高い精度を示している。特にデータセット Cora の Structurally missing のケースにおいては提案手法は比較手法から 33.38% も改善している。この要因は、観測値が少ないケースでは観測値 1 つ 1 つの情報が重要であるため、観測値を用いて欠損値を GCN に適した方法で補完している提案手法の精度が高くなったと考えられる。

また図 4.2 から 4.5 までは提案手法である GCN_recursive モデルの再帰回数によるノード分類の精度の変化を調べている。この図ではそれぞれ Cora,Citeseer,Amaphoto,Amacomp の 4 種類データセットで Uniform randomly missing, Biased randomly missing,Structually Missing の 3 種類の欠損パターンにおける欠損値が 0.2,0.5,0.8 での精度について示している。

全体的なグラフから、欠損率が 0.2 や 0.5 などあまり大きくない場合は再帰回数によって分類精度がほとんど変化しないことが分かる。この要因としては欠損率が少ない場合は観測値が多いために再帰回数が少なくても欠損値補完のためのアルゴリズムが収束するからであると考えられる。

一方、欠損率が 0.8 と大きい場合はデータセット Cora や Citeseer においては再帰回数が 16 以下などの少ない場合は再帰回数が 32 以上の場合に比べて精度が下がっている傾向があることが確認できた。データセット Amaphoto や Amacomp においては欠損率が 0.8 の場合でも再帰回数による精度の変化はほとんど見られなかった。この要因としてはデータセット Cora や Citeseer に比べ、Amaphoto や Amacomp は密なグラフであるため、ノードの平均次数が大きく欠損率が高い場合でも少ない再帰回数で特徴量が収束するからであると考えられる。

4.3 リンク予測

提案手法の評価を行うためにリンク予測の精度を調べた。実験はノード分類と同様に Uniform randomly missing, Biased randomly missing, Structurally missing の 3 種類の欠損パターンを用いた。また、欠損率は 10% から 90% までを 10% づつ変化させて 9 種類のパターンを用いた。また訓練/検証/テストエッジのデータセットの分割においては比較手法と同様に、全体のエッジから 10% を訓練エッジに、5% を検証エッジに、残りの 85% をテストエッジにランダムに選択し決定した。また、訓練

表 4.3　Cora データセットにおけるノード分類の精度

欠損パターン	欠損率	10%	20%	30%	40%	50%	60%	70%	80%	90%
Uniform Randomly Missing	MEAN	80.96	80.41	79.48	78.51	77.17	73.66	56.24	20.49	13.22
	K-NN	80.45	80.10	78.86	77.26	75.34	71.55	66.44	40.99	15.11
	MFT	80.70	80.03	78.97	78.12	76.43	71.33	45.82	27.22	23.98
	SoftImp	80.74	80.32	79.63	78.68	77.32	74.26	70.36	64.93	41.20
	MICE	–	–	–	–	–	–	–	–	–
	MissForest	80.68	80.43	79.74	79.27	76.12	73.70	68.31	60.92	45.89
	VAE	80.91	80.47	79.18	78.38	76.84	72.41	50.79	18.12	13.27
	GAIN	80.43	79.72	78.35	77.01	75.31	72.50	70.34	64.85	58.87
	GINN	80.77	80.01	78.77	76.67	74.44	70.58	58.60	18.04	13.19
	GCNmf	81.70	**81.66**	80.41	79.52	77.91	76.67	74.38	70.57	63.49
	GCN_neighbor	81.14	80.89	80.27	78.90	78.42	77.15	75.75	73.57	69.11
	GCN_recursive	**82.04**	81.64	**80.74**	**79.87**	**79.40**	**78.86**	**78.03**	**76.86**	**76.35**
Biased Randomly Missing	Mean	81.22	80.37	78.95	77.46	75.94	72.44	53.14	20.39	13.40
	K-NN	80.75	79.94	78.33	77.17	75.62	72.66	67.05	54.71	15.13
	MFT	80.75	75.01	56.28	55.76	43.81	29.31	25.88	21.79	21.07
	SoftImp	81.04	80.30	78.80	78.50	75.99	73.65	61.37	60.06	46.38
	MICE	–	–	–	–	–	–	–	–	–
	MissForest	80.90	80.10	78.79	77.54	74.66	71.04	65.28	56.65	44.30
	VAE	80.92	80.33	78.86	77.25	75.74	69.29	53.53	18.11	13.27
	GAIN	80.68	79.62	78.54	77.41	75.84	73.82	69.18	63.99	59.41
	GINN	80.86	80.10	78.45	76.80	74.60	72.08	65.72	50.08	13.22
	GCNmf	**82.29**	**81.09**	80.00	79.23	77.33	76.19	72.57	68.19	65.73
	GCN_neighbor	81.51	80.23	78.29	78.54	76.09	74.68	74.54	72.49	70.21
	GCN_recursive	81.31	80.58	**80.57**	**80.11**	**79.89**	**79.53**	**78.18**	**77.14**	**76.30**
Structurally Missing	MEAN	80.92	80.40	79.05	77.73	75.22	70.18	56.30	25.56	13.86
	K-NN	80.76	80.26	78.63	77.51	74.51	70.86	63.29	37.97	13.95
	MFT	80.91	80.34	78.93	77.48	74.47	69.13	52.65	29.96	17.05
	SoftImp	79.71	69.47	69.31	52.53	44.71	40.07	36.68	28.51	27.90
	MICE	80.92	80.40	79.05	77.72	75.22	70.18	56.30	25.56	13.86
	MissForest	80.48	79.88	78.54	76.93	73.88	68.13	54.29	30.82	14.05
	VAE	80.63	79.98	78.57	77.42	74.69	69.95	60.71	36.59	17.27
	GAIN	80.53	79.78	78.36	77.09	74.25	69.90	61.33	41.09	18.43
	GINN	80.85	80.27	78.88	77.35	74.76	70.58	59.45	29.15	13.92
	GCNmf	81.65	80.77	80.67	79.24	77.43	75.97	72.69	68.00	55.64
	GCN_neighbor	81.77	81.10	79.82	79.63	78.47	76.17	74.49	71.06	62.36
	GCN_recursive	**81.97**	**81.37**	**80.68**	**80.08**	**78.73**	**78.13**	**78.23**	**76.28**	**74.21**
GCN						81.49				
GCN ($\mathbf{X} = \mathbf{I}_N$)						63.22				

表 4.4　Citeseer データセットにおけるノード分類の精度

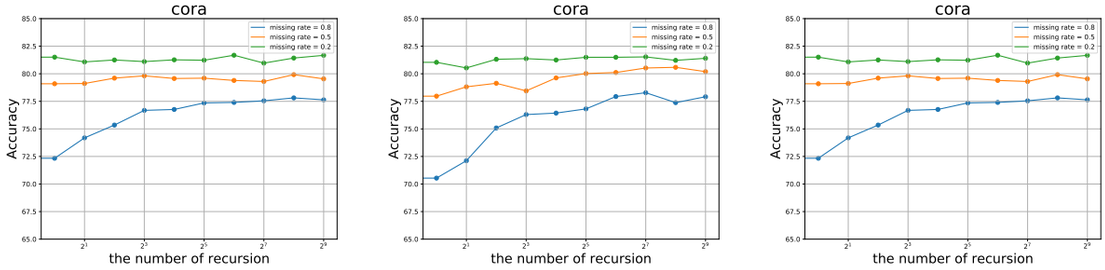
欠損パターン	欠損率	10%	20%	30%	40%	50%	60%	70%	80%	90%
Uniform Randomly Missing	MEAN	69.88	69.62	68.97	65.12	54.62	37.39	18.29	12.28	11.88
	K-NN	69.84	69.38	68.69	67.18	62.64	54.75	32.20	14.84	12.73
	MFT	69.70	69.51	68.74	65.31	60.56	41.53	34.10	17.26	19.29
	SoftImp	69.63	69.34	69.23	68.47	66.35	**65.53**	60.86	52.23	31.08
	MICE	–	–	–	–	–	–	–	–	–
	MissForest	–	–	–	–	–	–	–	–	–
	VAE	69.80	69.39	68.54	64.13	50.91	29.62	18.45	12.49	11.00
	GAIN	69.64	68.88	67.56	65.97	63.86	60.74	55.77	52.05	42.73
	GINN	70.07	69.79	68.87	68.14	63.21	43.61	20.74	13.26	11.31
	GCNmf	**70.93**	**70.82**	**69.84**	**68.83**	**67.03**	64.78	60.70	55.38	47.78
	GCN_neighbor	68.37	67.11	66.48	65.37	63.43	61.27	59.50	56.18	51.50
	GCN_recursive	69.34	68.82	68.25	66.71	65.21	64.28	**61.60**	**58.80**	**57.29**
Biased Randomly Missing	Mean	69.98	68.95	67.91	65.87	60.33	40.68	25.45	14.01	13.32
	K-NN	70.04	68.87	68.88	67.38	64.47	62.45	52.66	32.60	12.64
	MFT	69.88	67.68	63.17	45.49	25.99	20.22	20.82	18.53	18.30
	SoftImp	69.83	67.36	68.36	67.49	64.26	62.38	58.45	55.63	32.95
	MICE	–	–	–	–	–	–	–	–	–
	MissForest	–	–	–	–	–	–	–	–	–
	VAE	70.05	69.13	68.21	63.44	55.71	38.55	21.98	13.34	11.17
	GAIN	69.81	68.76	68.38	66.83	64.05	62.15	58.31	52.14	42.18
	GINN	69.96	69.60	69.63	68.67	64.93	62.14	55.01	31.37	12.91
	GCNmf	**71.01**	**69.99**	**69.96**	**68.89**	**66.30**	**64.67**	61.06	54.70	46.14
	GCN_neighbor	68.46	67.10	66.19	64.77	62.87	58.69	58.31	54.98	49.59
	GCN_recursive	68.76	68.14	67.98	64.88	63.85	63.44	**61.87**	**60.58**	**57.20**
Structurally Missing	MEAN	69.55	68.31	67.30	65.18	53.64	34.07	18.56	13.19	11.30
	K-NN	69.67	67.33	66.09	63.29	56.86	31.27	19.51	13.75	11.21
	MFT	69.84	68.21	66.67	63.02	51.08	34.29	16.81	14.34	15.75
	SoftImp	44.06	27.92	25.83	25.13	25.59	23.99	25.41	22.83	20.13
	MICE	–	–	–	–	–	–	–	–	–
	MissForest	–	–	–	–	–	–	–	–	–
	VAE	69.63	68.07	66.34	64.33	60.46	54.37	40.71	23.14	17.20
	GAIN	69.47	67.86	65.88	63.96	59.96	54.24	41.21	25.31	17.89
	GINN	69.64	67.88	66.24	63.71	55.76	40.20	18.63	13.23	12.32
	GCNmf	**70.44**	68.56	66.57	65.39	63.44	60.04	56.88	51.37	39.86
	GCN_neighbor	68.96	**68.59**	**67.45**	65.90	63.44	61.62	57.86	55.48	45.91
	GCN_recursive	69.18	68.51	66.90	**66.80**	**64.86**	**62.08**	**61.28**	**57.77**	**54.71**
GCN						70.65				
GCN ($\mathbf{X} = \mathbf{I}_N$)						40.55				

表 4.5　AmaPhoto データセットにおけるノード分類の精度

欠損パターン	欠損率	10%	20%	30%	40%	50%	60%	70%	80%	90%
Uniform Randomly Missing	MEAN	92.15	92.05	91.81	91.62	91.40	90.76	88.98	86.41	68.88
	K-NN	92.27	92.12	91.94	91.67	91.37	90.92	90.03	87.41	81.91
	MFT	92.23	92.07	91.88	91.51	91.15	90.11	88.28	85.17	75.73
	SoftImp	92.23	92.09	91.92	91.78	91.55	91.18	90.55	88.93	85.22
	MICE	92.23	92.07	91.97	91.75	91.52	91.22	90.42	86.43	82.88
	MissForest	92.18	92.09	91.82	91.61	91.42	90.71	89.17	86.03	82.82
	VAE	92.20	92.08	91.90	91.59	91.15	90.55	89.28	86.95	81.43
	GAIN	92.23	92.11	91.90	91.73	91.49	91.24	90.72	89.49	86.96
	GINN	92.25	92.03	91.87	91.53	91.14	90.56	88.59	85.02	79.80
	GCNmf	92.54	92.44	92.20	92.09	**92.09**	91.69	91.25	90.57	88.96
	GCN_neighbor	92.42	92.22	92.01	91.74	90.28	91.42	90.33	90.03	89.92
	GCN_recursive	**92.57**	**92.49**	**92.35**	**92.21**	91.96	**91.75**	**91.43**	**90.81**	**89.98**
Biased Randomly Missing	Mean	92.19	91.89	91.80	91.58	91.24	90.74	89.69	87.23	76.91
	K-NN	92.24	92.09	91.99	91.85	91.58	91.32	90.68	89.39	81.88
	MFT	92.17	92.03	91.98	91.71	91.40	90.99	89.89	87.46	75.14
	SoftImp	92.21	92.10	92.02	91.85	91.61	91.27	90.52	88.87	84.84
	MICE	92.16	92.06	92.00	91.76	91.58	91.24	90.54	88.64	82.45
	MissForest	92.16	92.09	92.07	91.81	91.35	90.67	89.77	86.85	82.72
	VAE	92.14	92.04	91.95	91.70	91.41	91.02	90.00	88.92	83.08
	GAIN	92.22	92.02	91.87	91.76	91.58	91.43	90.88	89.99	87.11
	GINN	92.24	92.04	91.95	91.78	91.48	91.16	90.40	88.35	79.18
	GCNmf	**92.72**	**92.69**	**92.55**	**92.61**	92.43	**92.33**	91.91	**91.58**	89.35
	GCN_neighbor	92.40	92.00	91.73	90.78	91.26	90.26	90.70	90.49	89.25
	GCN_recursive	92.60	92.53	92.52	92.52	**92.45**	92.20	**91.93**	91.42	**89.82**
Structurally Missing	MEAN	92.06	91.80	91.59	91.20	90.59	89.83	87.66	84.60	77.41
	K-NN	92.04	91.71	91.43	91.08	90.37	89.88	88.80	85.77	80.48
	MFT	92.08	91.83	91.59	91.18	90.56	89.80	87.58	84.36	77.69
	SoftImp	91.75	91.19	90.55	89.33	88.00	87.19	84.87	81.96	76.72
	MICE	92.05	91.87	91.59	91.24	90.60	89.86	87.82	84.57	77.32
	MissForest	92.04	91.70	91.42	91.15	90.49	90.07	88.81	85.51	75.35
	VAE	92.11	91.84	91.50	91.08	90.46	89.29	87.47	83.45	67.85
	GAIN	92.04	91.78	91.49	91.14	90.63	89.94	88.60	85.41	76.48
	GINN	92.09	91.83	91.53	91.16	90.43	89.61	87.77	84.53	77.14
	GCNmf	92.45	**92.32**	**92.08**	**91.88**	**91.52**	90.89	**90.39**	89.64	86.09
	GCN_neighbor	92.34	91.29	91.77	90.83	90.71	89.52	87.44	90.15	88.95
	GCN_recursive	**92.47**	92.18	91.98	91.55	91.43	**91.17**	89.79	**89.90**	**88.68**
GCN						92.35				
GCN ($\mathbf{X} = \mathbf{I}_N$)						88.77				

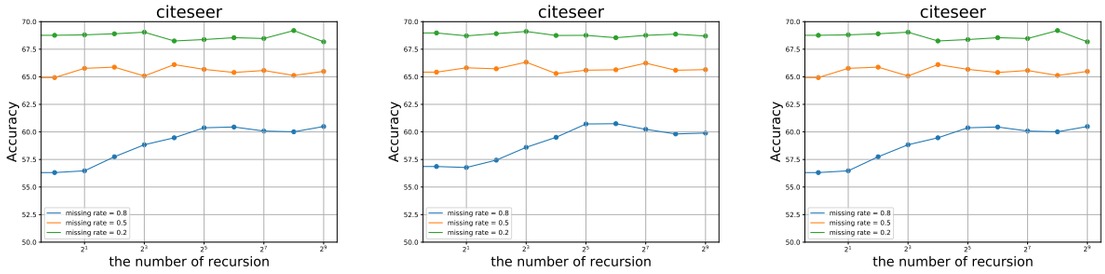
表 4.6　AmaComp データセットにおけるノード分類の精度

欠損パターン	欠損率	10%	20%	30%	40%	50%	60%	70%	80%	90%
Uniform Randomly Missing	MEAN	82.79	82.36	81.51	80.53	79.30	77.22	74.56	61.60	5.92
	K-NN	82.89	82.73	82.18	82.00	81.54	80.58	79.34	76.81	66.04
	MFT	82.82	82.54	82.05	81.58	80.76	79.28	77.11	72.31	49.42
	SoftImp	82.99	82.75	82.37	82.06	81.48	80.48	79.27	77.29	69.04
	MICE	82.83	82.76	82.43	82.28	81.66	80.59	78.63	75.00	63.60
	MissForest	–	–	–	–	80.89	79.57	78.22	76.00	71.98
	VAE	82.65	82.47	81.72	81.15	80.47	79.99	78.55	75.80	67.26
	GAIN	82.94	82.78	82.44	81.96	81.56	80.71	79.96	78.38	76.15
	GINN	82.94	82.78	82.27	81.65	80.89	78.53	76.46	73.24	58.34
	GCNmf	**86.32**	**86.07**	**85.98**	**85.77**	**85.46**	**84.94**	**84.03**	82.38	77.52
	GCN_neighbor	85.61	85.17	80.73	83.99	82.33	82.72	81.09	77.74	79.48
	GCN_recursive	85.83	85.60	85.27	85.14	84.71	84.67	83.00	**82.93**	**81.81**
Biased Randomly Missing	Mean	83.03	83.07	82.49	81.82	81.17	79.76	78.16	73.79	8.68
	K-NN	83.01	82.79	82.43	82.14	81.57	81.40	80.24	77.86	66.45
	MFT	82.98	82.86	82.39	81.93	81.30	80.18	78.66	74.96	50.53
	SoftImp	83.07	82.88	82.13	81.87	81.23	80.53	78.98	76.74	73.91
	MICE	83.07	82.77	82.44	81.94	81.56	80.84	79.40	76.71	64.11
	MissForest	–	–	81.88	–	80.52	79.62	78.27	76.66	71.74
	VAE	82.93	82.66	82.27	81.57	81.04	80.28	78.50	76.43	72.58
	GAIN	83.04	82.90	82.70	82.15	81.69	81.35	80.45	78.88	76.47
	GINN	83.10	82.71	82.58	81.94	81.63	80.81	79.29	76.53	58.18
	GCNmf	**86.41**	**86.35**	**86.27**	**86.16**	**85.83**	85.37	84.84	83.00	79.58
	GCN_neighbor	84.83	84.27	84.34	84.26	83.70	83.71	80.92	81.82	76.67
	GCN_recursive	85.75	85.82	85.52	85.38	85.05	**84.76**	**84.31**	**83.86**	**82.60**
Structurally Missing	MEAN	82.53	82.09	81.35	80.62	79.59	77.75	75.06	69.67	23.42
	K-NN	82.59	82.15	81.57	81.07	80.25	78.86	76.91	72.89	42.23
	MFT	82.48	81.91	81.43	80.58	79.40	77.64	75.19	69.97	27.33
	SoftImp	82.64	81.97	81.32	80.83	79.68	77.66	75.92	56.62	52.75
	MICE	82.71	82.13	81.51	80.62	79.36	77.35	74.57	67.59	45.07
	MissForest	82.65	82.20	81.84	81.04	79.18	78.66	75.98	71.91	12.05
	VAE	82.76	82.40	81.72	80.88	79.23	77.62	73.76	66.33	41.37
	GAIN	82.76	82.53	82.11	81.68	80.76	78.65	74.38	67.38	54.24
	GINN	82.55	82.10	81.46	80.75	79.59	77.67	75.08	70.40	26.10
	GCNmf	**86.37**	**86.22**	**85.80**	**85.43**	**85.24**	**84.73**	**84.06**	**80.63**	73.42
	GCN_neighbor	84.87	85.09	82.75	83.33	79.66	75.77	75.54	75.50	70.82
	GCN_recursive	85.85	85.24	85.00	84.43	83.90	83.91	82.05	79.99	**77.63**
GCN						82.94				
GCN ($\mathbf{X} = \mathbf{I}_N$)						81.60				



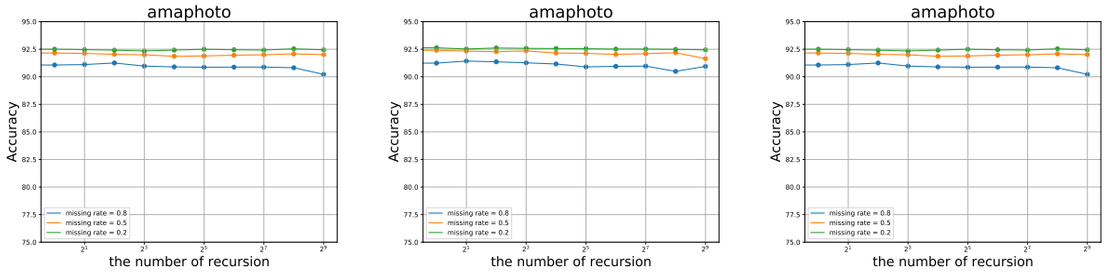
(a) Uniform randomly missing (b) Biased randomly missing (c) Structurally missing

図 4.2　Cora での分類精度



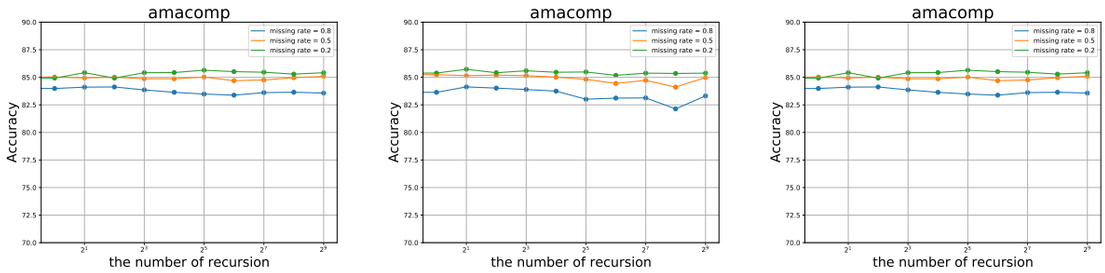
(a) Uniform randomly missing (b) Biased randomly missing (c) Structurally missing

図 4.3　Citeseer での分類精度



(a) Uniform randomly missing (b) Biased randomly missing (c) Structurally missing

図 4.4　Amaphoto での分類精度



(a) Uniform randomly missing (b) Biased randomly missing (c) Structurally missing

図 4.5　Amacomp での分類精度

エッジとテストエッジにおいては正例のエッジと等しい数の負例のエッジをエッジが存在しないノードペア間からランダムに選択した。また実験は 1 つの欠損パターンで 20 回の実験を行い、それを 5 つの欠損パターンの平均で精度を調べた。つまり合計で 100 回の実験の平均をリンク予測の精度とした。

リンク予測の指標はエッジの正例と負例の判定を ROC-AUC スコアとして求めた。この指標は 0 から 100 までの値をとり、値が大きいほど精度が良いことを示す。比較手法と同様の設定にするために、実験のエポック数を 200、学習率を 0.01 とした。また GCN を用いる手法において隠れ層の次元数は 32、潜在変数の次元を 16 に設定して実験を行った。また提案手法の 1 つである GCN_recursive モデルの再帰回数は 32 回として実験を行った。

表 4.7 と表 4.8 ではデータセット Cora,Citeseer におけるリンク予測の精度を示している。各セルには手法、欠損パターン、欠損率ごとの ROC-AUC スコアを示しており、太字の数字は、その欠損パターンと欠損率において最も精度が高いことを示している。提案手法は比較手法と比べると欠損率が低い場合はあまり精度が高くないといえる。しかし欠損率が高い場合は提案手法の 1 つである GCN_recursive モデルは比較手法と比べて優れた精度を達成していることが確認できた。具体的にはデータセット Cora においては全ての欠損パターンにおいて欠損率 90% の場合に提案手法が比較手法よりも高い精度を達成しており、さらにデータセット Citeseer においては Structurally missing の欠損パターンにおいて比較手法を上回っている。

半教師ありノード分類と異なり、提案手法が欠損率が低い場合に精度が低い理由は、提案手法が GCN に適した代入法を用いているために、リンク予測で扱う VGAE を用いたデコードに不向きな方法であるからであると考えられる。また提案手法 GCN_recursive が欠損率が高い場合に精度が高い理由は、グラフ構造を用いて再帰的に特徴量を補完しているため、欠損率が高くても欠損率が低い場合と比べて欠損している特徴量を補完する精度があまり下がらず、その結果リンク予測精度を維持できているからであると考えられる。

表 4.7　Cora データセットにおけるリンク予測の ROC-AUC スコア

欠損パターン	欠損率	10%	20%	30%	40%	50%	60%	70%	80%	90%
Uniform Randomly Missing	Mean	90.72	90.41	90.10	89.79	89.11	88.40	87.13	84.47	74.97
	K-NN	92.20	91.86	91.34	90.93	90.19	89.03	87.62	85.69	81.55
	MFT	92.16	91.86	91.37	90.91	90.14	88.37	86.11	84.10	79.94
	SoftImp	90.88	90.79	90.64	90.40	89.98	89.22	88.37	86.75	84.13
	MICE	–	–	–	–	–	–	–	–	–
	MissForest	92.32	92.04	91.61	90.95	90.33	89.34	88.33	86.41	82.78
	VAE	92.23	91.91	91.33	90.54	89.28	86.98	82.52	77.74	77.27
	GAIN	92.17	91.87	91.46	91.00	90.57	89.78	89.17	88.13	86.01
	GINN	92.15	91.96	91.62	91.00	90.28	88.94	87.66	84.73	74.90
	GCNmf	**94.09**	**93.50**	**93.05**	**92.40**	**92.29**	**91.79**	**90.77**	**88.32**	81.46
	GCN_neighbor	90.84	90.06	89.69	89.16	88.47	88.44	87.77	86.54	86.05
	GCN_recursive	90.41	90.53	90.06	89.45	89.68	88.07	88.65	87.38	**87.14**
Biased Randomly Missing	Mean	92.18	92.08	92.14	91.89	91.43	91.01	89.55	87.19	76.96
	K-NN	92.17	92.06	92.02	91.83	91.47	90.92	89.84	87.85	81.65
	MFT	92.17	91.44	90.65	90.00	89.50	88.91	87.48	85.36	80.20
	SoftImp	92.35	92.34	92.35	92.08	91.74	91.36	**90.03**	88.44	86.17
	MICE	–	–	–	–	–	–	–	–	–
	MissForest	92.27	92.22	92.22	91.80	91.22	90.34	88.90	86.86	83.58
	VAE	92.19	92.05	91.83	91.37	90.75	89.71	87.37	84.95	76.71
	GAIN	92.18	92.01	91.88	91.70	91.28	90.75	89.87	**88.75**	86.69
	GINN	92.15	92.11	92.04	91.88	91.52	90.89	89.45	87.36	75.38
	GCNmf	**94.35**	**94.20**	**93.90**	**93.15**	**92.43**	**91.46**	**90.03**	86.10	81.72
	GCN_neighbor	90.26	89.73	90.35	89.68	89.66	89.07	87.69	85.93	86.36
	GCN_recursive	90.00	89.53	88.85	89.18	88.79	87.58	87.13	87.29	**86.94**
Structurally Missing	Mean	90.34	89.79	89.12	88.26	87.12	85.33	83.23	79.61	71.79
	K-NN	91.60	91.08	90.38	89.36	88.34	87.16	85.40	82.09	76.12
	MFT	91.51	91.00	89.95	89.11	87.36	85.81	82.90	77.73	73.72
	SoftImp	90.29	89.67	88.86	87.86	86.77	85.36	83.07	81.53	77.38
	MICE	91.58	91.11	90.30	89.34	88.18	86.70	84.24	80.31	72.63
	MissForest	91.57	91.05	90.23	89.36	88.34	87.16	85.40	82.09	76.22
	VAE	91.49	90.76	89.49	87.27	83.81	80.07	73.46	67.55	65.80
	GAIN	91.60	91.08	90.38	89.36	88.34	87.16	85.40	82.09	76.12
	GINN	91.51	90.85	89.68	87.34	83.23	76.22	66.55	63.88	64.91
	GCNmf	**93.55**	**92.65**	**91.68**	**90.55**	**88.54**	86.19	81.96	76.35	67.86
	GCN_neighbor	91.03	89.74	89.55	88.73	87.90	86.71	85.47	84.12	79.78
	GCN_recursive	89.52	89.40	89.01	88.44	88.47	**87.26**	**87.30**	**86.66**	**85.20**
GCN						92.42				
GCN ($\mathbf{X} = \mathbf{I}_N$)						85.90				

表 4.8　Citeseer データセットにおけるリンク予測の ROC-AUC スコア

欠損パターン	欠損率	10%	20%	30%	40%	50%	60%	70%	80%	90%
Uniform Randomly Missing	MEAN	89.01	88.56	88.01	87.33	86.42	85.30	83.77	81.43	75.47
	K-NN	90.00	89.60	89.10	88.34	87.32	85.68	83.39	81.16	78.60
	MFT	89.86	89.43	88.81	87.72	85.76	83.24	81.20	79.97	77.94
	SOFTIMP	90.19	90.15	89.81	89.55	88.97	88.17	86.80	84.99	81.66
	MICE	–	–	–	–	–	–	–	–	–
	MISSFOREST	–	–	–	–	–	–	–	–	–
	VAE	89.85	89.09	88.13	87.22	85.36	83.55	80.64	74.89	64.69
	GAIN	89.96	89.53	89.07	88.36	87.51	86.52	85.35	83.93	81.70
	GINN	90.02	89.64	89.04	87.91	86.56	84.64	83.32	81.82	77.19
	GCNMF	**93.20**	**92.96**	**92.30**	**92.19**	**90.45**	**90.08**	**88.91**	**87.28**	**83.68**
	GCN_NEIGHBOR	89.00	87.14	86.02	85.20	84.32	83.48	82.92	81.51	81.36
	GCN_RECURSIVE	88.23	87.47	87.17	86.20	86.65	85.12	83.54	83.26	81.70
Biased Randomly Missing	MEAN	89.94	89.88	89.63	89.33	89.25	88.55	87.57	85.28	78.23
	K-NN	90.00	89.98	89.81	89.54	89.31	88.52	87.47	84.97	78.85
	MFT	89.98	87.50	85.88	85.07	84.32	83.76	82.85	81.54	78.23
	SOFTIMP	90.31	90.25	90.23	89.99	89.90	89.03	87.12	85.96	80.63
	MICE	–	–	–	–	–	–	–	–	–
	MISSFOREST	–	–	–	–	–	–	–	–	–
	VAE	89.90	89.25	88.33	87.32	86.26	83.78	83.05	80.71	62.51
	GAIN	89.97	89.87	89.60	89.32	88.89	87.85	87.00	85.05	**81.95**
	GINN	90.27	89.99	89.85	89.47	89.10	88.15	87.21	84.47	76.83
	GCNMF	**93.53**	**93.38**	**92.81**	**92.48**	**91.68**	**91.25**	**89.54**	**86.73**	81.43
	GCN_NEIGHBOR	88.36	87.67	86.81	85.71	85.39	83.40	83.17	81.71	80.20
	GCN_RECURSIVE	89.53	88.31	87.47	87.16	85.60	84.27	84.58	82.21	80.89
Structurally Missing	MEAN	88.16	86.95	85.76	84.20	82.43	80.83	78.92	75.79	69.76
	K-NN	89.50	88.36	87.01	85.52	83.85	82.11	79.81	76.49	70.86
	MFT	89.24	87.96	86.53	84.76	83.19	80.67	78.35	75.97	72.64
	SOFTIMP	89.50	88.36	87.01	85.52	83.85	82.11	79.81	76.49	70.86
	MICE	–	–	–	–	–	–	–	–	–
	MISSFOREST	–	–	–	–	–	–	–	–	–
	VAE	88.57	86.83	84.32	80.96	77.49	74.01	67.84	63.06	60.39
	GAIN	89.50	88.36	87.01	85.52	83.85	82.11	79.81	76.49	70.86
	GINN	87.48	83.35	77.50	70.06	64.31	59.45	57.95	54.88	50.81
	GCNMF	**92.23**	**90.54**	**88.77**	**85.74**	**84.78**	**84.59**	**82.00**	77.21	73.31
	GCN_NEIGHBOR	88.27	87.06	86.18	84.89	82.93	80.78	79.93	79.03	74.37
	GCN_RECURSIVE	88.45	87.38	86.60	84.61	82.94	81.97	81.03	**79.04**	**79.40**
GCN						90.25				
GCN ($\mathbf{X} = \mathbf{I}_N$)						79.94				

第 5 章

結論

本研究では欠損値を含むグラフデータに対して GCN(グラフ畳み込みネットワーク) を適用するために、グラフ構造を用いて欠損値を代入し補完する手法を提案した。提案手法としてノードの特徴量を近隣ノードの欠損していない特徴量の平均により補完する GCN_neighbor モデルと、近隣ノードの特徴量を用いて再帰的に補完する GCN_recursive モデルを定義し実験を行った。実験において提案手法の GCN は 2 層 GCN を用い勾配降下法で学習するモデルを用いた。実験ではノード分類とリンク予測のタスクを欠損率を 10% から 90% まで変化させたときの精度を調べた。実験の結果、多くのデータセットや欠損率で GCN_recursive モデルが比較手法よりも高い精度を示すことが確認できた。また欠損率が高い場合でも提案手法は比較手法と比べて高い精度を維持するモデルであることが確認できた。

本研究の今後の課題として以下の 3 つが挙げられる。

1 つ目の課題は欠損値補完に全ての特徴量の情報を用いていないことである。提案手法での欠損値補完のアルゴリズムは、特徴量の種類ごとにグラフ構造を用いて補完している。そのため、あるノードの欠損値補完はそのノードの別の特徴量に依存するものではない。この問題はそれぞれのノードごとの特徴量を共起しつつ欠損値補完をするモデルを作成することで解決できると考えられる。

2 つ目の課題は欠損値補完の異なる方法の模索である。本研究での 2 つの提案手法はいずれも近隣ノードの特徴量の平均を用いて欠損値補完したものである。しかし近隣ノードの特徴量に偏りをかけて学習する方法や Attention 機構を用いた方法、ノードエンベディングを用いてベクトル空間を用いて欠損値補完をする方法など様々なアプローチが考えられる。また異なるアプローチを組み合わせることにより精度のさらなる向上が見込めると思われる。

3 つ目の課題は異なる欠損パターンにおける実験である。本研究の Uniform randomly missing, Biased randomly missing, Structurally missing の 3 つの欠損パターンは全て人工的に作成させられたものであり MCAR に分類される。そのため MNAR や MAR といった偶発的に生じた欠損パターンを考慮したモデルを考えることも今後の課題の 1 つである。

付録 A

比較手法のパラメータ設定

比較手法で用いたハイパーパラメータについて述べる。比較手法の精度は [46] の値を用いた。詳細なハイパーパラメータとしては以下の通り設定した。

- MEAN [13]:
  デフォルトのモデルを用いている。
- K-NN [2]:
  ノードごとに考慮する近傍ノード数は $k = 5$ である。
- MFT [25]:
  行列変換で用いる隠れ層のランク数は 10 である。
- SoftImp [30]:
  デフォルトのモデルを用いている。
- MICE [7]:
  デフォルトのモデルを用いている。
- MissForest [44]:
  デフォルトのモデルを用いている。
- VAE [22]:
  エンコーダ, デコーダ共に 2 層パーセプトロンを用いており、隠れ層の次元数が 32、潜在表現の次元数が 16、ドロップアウト率が 0.1 である。
- GAIN [53]:
  ヒント率が 0.9 であり、損失関数でのトレードオフパラメータは $\alpha = 10$ である。
- GINN [43]:
  デフォルトのモデルを用いている。
- GCNmf [46]:
  GMM の正規分布数は $K = 5$ である。

謝辞

本研究を進めるにあたり、指導教員である村田剛志教授には丁寧なご指導をいただきました。また研究室の先輩である田口響氏には多くの助言をいただきました。さらに村田研究室の皆様には貴重な意見をいただきました。ここに感謝の意を表します。

参考文献

[1] Sami Abu-El-Haija, Bryan Perozzi, Amol Kapoor, Nazanin Alipourfard, Kristina Lerman, Hrayr Harutyunyan, Greg Ver Steeg, and Aram Galstyan. Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *international conference on machine learning*, pp. 21–29. PMLR, 2019.

[2] Gustavo EAPA Batista, Maria Carolina Monard, et al. A study of k-nearest neighbour as an imputation method. *HIS*, Vol. 87, No. 48, pp. 251–260, 2002.

[3] Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

[4] Mikhail Belkin, Irina Matveeva, and Partha Niyogi. Regularization and semi-supervised learning on large graphs. In *International Conference on Computational Learning Theory*, pp. 624–638. Springer, 2004.

[5] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of machine learning research*, Vol. 7, No. 11, 2006.

[6] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

[7] S van Buuren and Karin Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in r. *Journal of statistical software*, pp. 1–68, 2010.

[8] Hongyun Cai, Vincent W Zheng, and Kevin Chen-Chuan Chang. A comprehensive survey of graph embedding: problems, techniques and applications. *IEEE Transactions on Knowledge and Data Engineering*, Vol. 30, No. 9, pp. 1616–1637, 2018.

[9] Jie Chen, Tengfei Ma, and Cao Xiao. FastGCN: Fast learning with graph convolutional networks via importance sampling. In *Proceedings of ICLR*, 2018.

[10] Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In *Proceedings of KDD*, pp. 257–266, 2019.

[11] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. In *Proceedings of NeurIPS*, pp. 2224–2232, 2015.

[12] Linton Freeman. The development of social network analysis. *A Study in the Sociology of Science*, Vol. 1, No. 687, pp. 159–167, 2004.

[13] Pedro J García-Laencina, José-Luis Sancho-Gómez, and Aníbal R Figueiras-Vidal. Pattern classification with missing data: a review. *Neural Computing and Applications*, Vol. 19, No. 2, pp. 263–282, 2010.

[14] Lise Getoor and Christopher P Diehl. Link mining: a survey. *Acm Sigkdd Explorations Newsletter*, Vol. 7, No. 2, pp. 3–12, 2005.

[15] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In *Proceedings of ICML*, pp. 1263–1272, 2017.

[16] 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*, pp. 1263–1272. PMLR, 2017.

[17] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of KDD*, pp. 855–864, 2016.

[18] William L Hamilton. Graph representation learning. *Synthesis Lectures on Artifical Intelligence and Machine Learning*, Vol. 14, No. 3, pp. 1–159, 2020.

[19] William L Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of NeurIPS*, pp. 1025–1035, 2017.

[20] Kai Jiang, Haixia Chen, and Senmiao Yuan. Classification for incomplete data using classifier ensembles. In *Proceedings of ICNNB*, pp. 559–563, 2005.

[21] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of ICLR*, 2015.

[22] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. pp. 1–14, 2014.

[23] Thomas N Kipf and Max Welling. Variational graph auto-encoders. *NIPS Workshop on Bayesian Deep Learning*, 2016.

[24] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *Proceedings of ICLR*, 2017.

[25] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, Vol. 42, No. 8, pp. 30–37, 2009.

[26] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, Vol. 25, pp. 1097–1105, 2012.

[27] Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of AAAI*, 2018.

[28] Roderick JA Little and Donald B Rubin. *Statistical analysis with missing data*. John Wiley & Sons, 2002.

[29] Sitao Luan, Mingde Zhao, Xiao-Wen Chang, and Doina Precup. Break the ceiling: Stronger multi-scale deep graph convolutional networks. In *Proceedings of NeurIPS*. 2019.

[30] Rahul Mazumder, Trevor Hastie, and Robert Tibshirani. Spectral regularization algorithms for learning large incomplete matrices. *J. Mach. Learn. Res.*, Vol. 11, pp. 2287–2322, 2010.

[31] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of SIGIR*, pp. 43–52, 2015.

[32] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pp. 3111–3119, 2013.

[33] Zhewei Wei Ming Chen, Bolin Ding Zengfeng Huang, and Yaliang Li. Simple and deep graph convolutional networks. In *Proceedings of ICML*, 2020.

[34] Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodola, Jan Svoboda, and Michael M Bronstein. Geometric deep learning on graphs and manifolds using mixture model

cnns. In *Proceedings of CVPR*, pp. 5115–5124, 2017.

[35] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of KDD*, pp. 701–710, 2014.

[36] Leonardo FR Ribeiro, Pedro HP Saverese, and Daniel R Figueiredo. struc2vec: Learning node representations from structural identity. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 385–394, 2017.

[37] Donald B Rubin. *Multiple imputation for nonresponse in surveys*, Vol. 81. John Wiley & Sons, 2004.

[38] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, Vol. 20, No. 1, pp. 61–80, 2008.

[39] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI Magazine*, Vol. 29, No. 3, p. 93, 2008.

[40] David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE signal processing magazine*, Vol. 30, No. 3, pp. 83–98, 2013.

[41] Marek Śmieja, Łukas Struski, Jacek Tabor, and Mateusz Marzec. Generalized rbf kernel for incomplete data. *Knowledge-Based Systems*, Vol. 173, pp. 150–162, 2019.

[42] Marek Śmieja, Łukasz Struski, Jacek Tabor, Bartosz Zieliński, and Przemysław Spurek. Processing of missing data by neural networks. In *Proceedings of NeurIPS*, pp. 2719–2729, 2018.

[43] Indro Spinelli, Simone Scardapane, and Uncini Aurelio. Missing data imputation with adversarially-trained graph convolutional networks. *Neural Networks*, Vol. 129, pp. 249–260, 2020.

[44] Daniel J. Stekhoven and Peter Bühlmann. MissForest—non-parametric missing value imputation for mixed-type data. *Bioinformatics*, Vol. 28, No. 1, pp. 112–118, 2011.

[45] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deep convolutional network cascade for facial point detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3476–3483, 2013.

[46] Hibiki Taguchi, Xin Liu, and Tsuyoshi Murata. Graph convolutional networks for graphs containing missing features. *Future Generation Computer Systems*, Vol. 117, pp. 155–168, 2021.

[47] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *Proceedings of WWW*, pp. 1067–1077, 2015.

[48] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *Proceedings of ICLR*, 2018.

[49] Daixin Wang, Peng Cui, and Wenwu Zhu. Structural deep network embedding. In *Proceedings of KDD*, pp. 1225–1234, 2016.

[50] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–21, 2020.

[51] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *Proceedings of ICLR*, 2018.

[52] Rex Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer:

Generating explanations for graph neural networks. In *Proceedings of NeurIPS*, 2019.

[53] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. GAIN: Missing data imputation using generative adversarial nets. In *Proceedings of ICML*, Vol. 80, pp. 5689–5698, 2018.

[54] Xiang Zhang and Marinka Zitnik. Gnnguard: Defending graph neural networks against adversarial attacks. In *Proceedings of NeurIPS*, 2020.

[55] Dengyong Zhou, Olivier Bousquet, Thomas N Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. In *Advances in neural information processing systems*, pp. 321–328, 2004.

[56] T. Zhou, J. Ren, M. Medo, and Y. C. Zhang. Bipartite network projection and personal recommendation. *Phys. Rev. E*, Vol. 76, p. 046115, 2007.

[57] Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International conference on Machine learning (ICML-03)*, pp. 912–919, 2003.