グラフ予測のための二部グラフ最大重みマッチングに基づくノードの分散表現学習

山崎　翔平† 　原田　圭† 　佐々木勇和† 　鬼塚　真†

† 大阪大学大学院情報科学研究科　〒 565–0871 大阪府吹田市山田丘 1-5

E-mail: †{yamasaki.shohei,harada.kei,sasaki,onizuka}@ist.osaka-u.ac.jp

あらまし　属性付き時系列グラフを対象としたノード属性予測やリンク予測等の既存のグラフ予測タスクの多くは閉世界仮説を仮定しており，将来新たに出現する未知ノードの出現は考慮していない．本稿では，開世界仮説に基づく属性付き時系列グラフにおける未知ノードの分散表現学習を行うフレームワークを提案する．提案フレームワークは予測された未知ノード集合と真の未知ノード集合を対とする二部グラフを構築し，最大重みマッチング利得を最大化するように未知ノード集合の分散表現を学習する．本手法の入出力データと教師データは順序を持たないノード集合であるため，点群のためのネットワークアーキテクチャである PointNet を採用する．実験では人工データを用いて，グラフ規模と二部マッチング利得の関係性を明らかにし，本手法が未知ノードの分散表現学習に有効であることを示す．

キーワード　時系列グラフ，分散表現学習，二部マッチング

1. はじめに

グラフ構造は現実世界の複雑な現象をモデル化する一般的なデータ表現である．グラフ構造で表現されるデータとしては論文の引用ネットワークやソーシャルネットワークが挙げられる．実世界に存在するグラフ構造の多くはノードが属性を有し，また時間変化によってグラフ構造が変化する．このような時間変化を伴う属性付きのグラフは，属性付き時系列グラフと呼ばれる．例えば，論文の引用ネットワークの場合，ノードを論文，リンクを引用関係，属性値を論文タイトルとすれば，論文数や引用関係は年々変化していくため，属性付き時系列グラフとしてモデル化できる．SNS や IoT の普及により属性付き時系列グラフとしてモデル化できる対象はますます増加傾向にあり，時間変化する属性付き時系列グラフの将来の構造を予測する技術は多様な分野への応用が期待できるとされ大きな注目を集めている．特に，将来のグラフにおけるノード属性を予測する属性予測や，リンク関係を予測するリンク予測は実世界での利用性の高さから多くの技術が提案されている [1–3]．

これまでの多くの研究において属性予測やリンク予測等の既存のグラフ予測タスクの多くは暗黙的に過去に観測されたノードのみを予測の対象としており，将来新たに出現するノード（以下，未知ノード）の属性やリンクは予測していない．既存の属性予測やリンク予測では共通して閉世界仮説（CWA: Closed World Assumption）[4] を仮定している．CWA とは，現時点で真であると判明していない事象については偽とみなす仮説であり，将来新たに追加される事象は考慮しない．つまり，未知ノードの属性予測やリンク予測を行わないため，CWA は静的グラフにおいて有効な仮説であり，動的グラフには適していない．例えば，論文の引用ネットワークでは，毎年新たな論文が追加される．既出の論文が何の論文から引用されるかを予測するリンク予測において，新たな投稿論文を想定しないのは現実的ではない．このように，実世界の多くのグラフは動的に変化し，絶えず新たなノードが追加されるため，属性付き時系列グラフを対象とした将来のグラフ予測タスクに CWA を仮定するのは不適切であると言える．

そこで本稿では，確率的データベース [5] と知識ベース [6] から開世界仮説（OWA: Open World Assumption）の概念をグラフ予測に取り入れることで，未知ノードの出現を考慮した将来予測タスクに取り組む．OWA とは現時点で真であると判明していない事象については未知とする仮説であり，将来新たに追加される事象も考慮に入れる．OWA においては，未知ノードの属性やそれに接続されるリンクも予測対象であるため，未知ノード集合の予測も同時に行う．実世界に存在する属性付き時系列グラフの多くは属性変化，リンク変化だけでなくノード集合の変化を伴う．そのため，OWA を仮定してグラフ予測を行うことで，CWA を前提とした属性予測やリンク予測と比較してより多くの分野へ応用できることが期待される．そこで本稿では，OWA を仮定したグラフ予測として，未知ノードの分散表現学習を行うタスクに取り組む．一般的にノードの分散表現を得ることで，ノード集合に対してベクトル空間に基づく機械学習手法を適用できるようになり，属性予測やリンク予測等の多様なタスクに応用可能となる．

未知ノードの分散表現を得ることは容易ではなく，筆者らの知る限り既存研究も存在しない．例えば，未知ノードの分散表現を獲得するモデルとして，過去に観測されたノード集合，リンク集合およびノード属性を学習し，未知ノード集合とその分散表現を出力する教師あり学習モデルが考えられる．このモデルは訓練時に，出力データ（予測されたノード集合）と正解データ（真のノード集合）の比較を行う必要があるが，モデル出力は順序を持たない集合であるため，予測されたノード集合と真のノード集合の対応関係は明らかでない．訓練時における予測されたノード集合と真のノード集合の対応関係はモデルが

予測する未知ノードの分散表現に影響を及ぼすため，慎重に決定される必要がある．

そこで本稿では，過去に観測された属性付き時系列グラフを用いて将来出現する未知ノード集合とその分散表現を予測するフレームワーク Deep MatchMax を提案する．Deep MatchMax は，予測されたノード集合と真のノード集合を対とする二部グラフを構築し，予測されたノード集合と真のノード集合の対応関係，および最大重みマッチング利得を最大化するようなノード集合の分散表現を学習する．ネットワークの入出力データと教師データは順序を持たないノード集合であり，最大重みマッチングから得られる二部グラフのリンク関係は学習のイテレーションごとに変化する．このような性質を持つ属性付きのノード集合は点群の性質に類似するため，Deep MatchMax のメインアーキテクチャとして点群の学習に有用な PointNet [7] を採用する．これにより，予測されたノード集合と真のノード集合の対応関係が決定し，さらに未知ノードの分散表現を獲得できる．

実験では，人工データを用いて本手法の有効性を検証する．まず，ノード属性が正規分布に従う人工データを用いて，グラフ規模と二部グラフにおける最大重みマッチング利得の関係性を明らかにする．次に，ベースライン手法と比較を行い，本手法が未知ノードの分散表現学習に有効であることを示す．

本稿の構成は以下の通りである．2 章で事前知識を述べる．3 章では本稿で取り組む問題定義について説明し，4 章で提案フレームワークについて述べる．5 章で評価実験について説明する．6 章で関連研究について述べ，最後に 7 章で本稿をまとめる．

2. 事前知識

本章では，2.1 節で本稿で用いる表記の定義を行う．Deep MatchMax では入力ノード集合の順序が変わっても，出力結果が変わらないことを求められる．出力結果とは予測されたノード集合と真のノード集合の対応関係，および未知ノードの分散表現のことである．このような順序についての不変性を要件に持つ他のタスクとして点群分類/セグメントタスクが存在する．2.2 節では，点群分類/セグメントタスクに適用可能な PointNet について説明する．

2.1 表　　記

本稿では各ノードが属性を持つ属性付きグラフを対象としており，時系列グラフをノード，リンク，ノード属性が時間変化するグラフと定義する．以下に詳細な定義を行う．

定義 1 (属性付き時系列グラフ)　ノード集合 V，リンク集合 $E \subseteqq V \times V$，および属性行列 $X \in \mathbb{R}^{|V| \times n}$ の三つ組 $G := (V, E, X)$ を属性付きグラフと呼ぶ．これらのどれかが時間によって変化する場合，G を属性付き時系列グラフと呼び，タイムステップ t における属性付き時系列グラフを $G_t := (V_t, E_t, X_t)$ と表記する．また，ノード $v_t \in V_t$ の分散表現は $\mathrm{vec}(v_t) \in \mathbb{R}^d$ と表記する．

提案フレームワーク Deep MatchMax は二部グラフを構築し最大重みマッチング利得を最大化するように学習を行う．ここでは二部グラフと最大重みマッチングの定義を行う．

定義 2 (二部グラフ)　無向グラフ $G = (V, E)$ において，ノード集合 V を互いに素な 2 つの部分集合 P と Q に分割したとき，リンク集合が $E \subseteqq P \times Q$ を満たすとき G を二部グラフと呼び，形式的に $G := ((P, Q), E)$ と表記する．特に，$E = P \times Q$ を満たすとき G を完全二部グラフと呼び，$K_{|P|,|Q|}$ と表記する．

定義 3 (最大重みマッチング)　無向グラフ $G = (V, E)$ において，部分リンク集合 $M \subseteqq E$ の中で，M のどの 2 つリンク関係をとっても同じノードに接続されないとき，M を G のマッチングと呼ぶ．リンク関係 $\forall e \in E$ に対して $w(e) \geqq 0$ となるような非負重み関数 $w : E \to \mathbb{R}$ を考える．G のマッチング $M \subseteqq E$ の中で，G の任意のマッチング M' に対して，$\sum_{e \in M} w(e) \geqq \sum_{e \in M'} w(e)$ を満たすとき，M を w に関する G の最大重みマッチングという．このとき，$\sum_{e \in M} w(e)$ を最大重みマッチング利得と呼び，$\mathrm{Gain}(M, w)$ と表記する．

本稿ではこれまでに観測していない新たな未知ノードの集合を予測する．時系列グラフにおける観測ノード集合および未知ノードの定義を述べる．

定義 4 (観測ノード集合)　属性付き時系列グラフを G_t とする．タイムステップ t から L ステップ前までに観測された属性付き時系列グラフのノード集合の集合を観測ノード集合と呼び，$\mathcal{V}_{t,L} := \bigcup_{i=t-L+1}^{t} V_i$ と表記する．

定義 5 (未知ノード)　タイムステップ t におけるノードの集合 $V_t \setminus \mathcal{V}_{t-1,L}$ を未知ノード集合と呼び，U_t と表記する．$U_t \neq \emptyset$ であるとき，ノード $v \in U_t$ を未知ノードと呼ぶ．

Deep MatchMax は予測された未知ノード集合と真の未知ノード集合の対応関係を決定する．以降では，教師あり学習モデルの出力には $\hat{}$ を付与することで，出力データと正解データを区別する．例えば，タイムステップ $t+1$ の予測されたノード集合を $\hat{V}_{t+1}$，真のノード集合を V_{t+1} と表記する．

2.2 PointNet

PointNet は Charles ら [7] によって提案された点群分類/セグメントタスクに取り組む深層学習フレームワークである．点群分類/セグメントタスクには入力データの順序を変更しても同じ結果を出力できなければならない（**順序についての不変性**）．以下では PointNet が順序についての不変性を満たすために採用するアプローチを説明する．

点群を $x_1, \ldots, x_n \in \mathbb{R}^N$ とする．このような入力点群データの順序を変更しても同じ結果を出力する関数として，以下の集合関数

$$f(\{x_1, \ldots, x_n\}),\ f : 2^{\mathbb{R}^N} \to \mathbb{R} \qquad (1)$$

を考えることができる．この関数の実装方法として，PointNet

では対称関数を使用して点群の情報を集約するというアプローチを採用している．式（1）の集合関数を以下の関数の組み合わせで近似する．

$$f(\{x_1,\ldots,x_n\}) \approx g(h(x_1),\ldots,h(x_n)) \qquad (2)$$

ただし，$h:\mathbb{R}^N \to \mathbb{R}^K$ であり，$g:\underbrace{\mathbb{R}^K \times \mathbb{R}^K \times \cdots \times \mathbb{R}^K}_{n} \to \mathbb{R}$ は対称関数である．PointNet では h に多層パーセプトロンを，g の対称関数として MaxPooling 関数を使用している．

3. 問題定義

本稿では，未知ノード集合の予測とその要素が持つ分散表現の予測に取り組む．一般的にノード分散表現はリンク予測等の応用タスクを解くために機械学習モデルに適用される．そのモデルが教師あり学習を伴う場合やモデルを評価する場合，モデル出力データと正解データの比較や誤差の計算を行う．そのため，予測された未知ノードの分散表現が決まる時，それに対応する真の未知ノードが何であるかが決定されている必要がある．そこで，分散表現の学習と同時に予測されたノード集合と真のノード集合のマッチング問題に取り組む．問題定義を以下に示す．

問題定義 (未知ノード集合予測と分散表現学習) タイムステップ t から L ステップ前までの一連の属性付き時系列グラフ $\{G_{t-L+1}, G_{t-L+2}, \ldots, G_t\}$ が与えられたとき，タイムステップ $t+1$ における未知ノード集合 U_{t+1} を予測する．また，以下の式（3）で与えられるように，最大重みマッチング利得を最大化する未知ノード $\forall \hat{v} \in \hat{U}_{t+1}$ の分散表現 $\mathrm{vec}(\hat{v})$ を予測する．

$$\text{maximize} \quad \mathrm{Gain}(M,w) = \sum_{(\hat{v},v)\in M} w((\hat{v},v)) \qquad (3)$$

ただし，M は式（4）の非負重み関数 w に関する完全二部グラフ K の最大重みマッチングである．

$$w((\hat{v},v)) = \frac{\mathrm{vec}(\hat{v})\cdot\mathrm{vec}(v)}{|\mathrm{vec}(\hat{v})||\mathrm{vec}(v)|},\ w: E \to \mathbb{R} \qquad (4)$$

$$K = ((\hat{U}_{t+1}, U_{t+1}), E),\ E = \hat{U}_{t+1} \times U_{t+1}$$

この問題を解くことで，未知ノード $\forall \hat{v} \in \hat{U}_{t+1}$ の分散表現 $\mathrm{vec}(\hat{v})$ と予測されたノード集合と真のノード集合の対応関係 M が決まる．これにより，他の機械学習手法を用いた未知ノードの教師あり学習が可能となる．

4. Deep MatchMax

本章では提案フレームワーク Deep MatchMax について詳細を述べる．

4.1 設計方針

提案フレームワークはタイムステップ t から L ステップ前までの一連の属性付き時系列グラフ $\{G_{t-L+1}, G_{t-L+2}, \ldots, G_t\}$ を入力とし，タイムステップ $t+1$ における未知ノード集合 $\hat{U}_{t+1}$ とその要素が持つ分散表現 $\mathrm{vec}(v),\ \forall v \in \hat{U}_{t+1}$ を出力する．また，これと同時に予測された未知ノード集合 $\hat{U}_{t+1}$ と真の未知ノード集合 U_{t+1} の対応関係を決定する．これを達成するために以下の機能が必要である．

（1） 未知ノード集合 $\hat{U}_{t+1}$ を予測

（2） 最大重みマッチング利得の最大化

それぞれの機能は行う予測・操作の性質が異なる．機能（1）は入力グラフの統計量から未知ノード集合 U_{t+1} の要素数を予測し，各ノードに分散表現の初期値を付与する．機能（2）では，予測された未知ノード集合 $\hat{U}_{t+1}$ と真の未知ノード集合 U_{t+1} から成る二部グラフの最大重みマッチング利得を最大化するような分散表現 $\mathrm{vec}(v),\ \forall v \in \hat{U}_{t+1}$ を予測する．

機能（1）の操作を前処理とし，**Deep MatchMax** にて，機能（2）の予測結果を出力する．前処理については，4.2 節で詳細を説明する．Deep MatchMax では順序についての不変性に対応する必要がある．例えば，$\{v_A, v_B, v_C\} \in \hat{U}_{t+1}$，$\{v_D, v_E, v_F\} \in U_{t+1}$ において，入力系列 (v_A, v_B, v_C) が与えられた時，出力されるマッチングが $\{(v_A, v_D), (v_B, v_E), (v_C, v_F)\}$ であるとする．この時，各ノードの分散表現が同じならば，入力系列が (v_C, v_A, v_B) や (v_B, v_C, v_A) であったとしても出力は $\{(v_A, v_D), (v_B, v_E), (v_C, v_F)\}$ でなければならない．つまり，集合の入力 $\{v_A, v_B, v_C\}$ に対してただ一つの結果 $\{(v_A, v_D), (v_B, v_E), (v_C, v_F)\}$ を出力する集合関数を実装する必要がある．Deep MatchMax では PointNet をメインアーキテクチャに採用することで順序の不変性に対応する．

4.2 未知ノード集合の予測

この節では Deep MatchMax の前処理として，未知ノード集合の予測について説明する．まず，入力グラフの観測ノード集合 $\mathcal{V}_{t,L}$ から，定義 5 に従い，各タイムステップにおける未知ノード集合 $\{U_{t-L+1}, U_{t-L+2}, \ldots, U_t\}$ を抽出し，それぞれの要素数をカウントする．この要素数の系列からタイムステップ $t+1$ における未知ノード集合の要素数を予測する．

6. 章で示すように，真の未知ノードの分散表現の獲得手法については隣接性と属性を考慮したものが多数提案されているが，本稿では簡単のために真の未知ノード $v \in U_{t+1}$ の分散表現は単に $\mathrm{vec}(v) := X_t[v]$ とする．これに合わせて，予測された未知ノードの分散表現の初期値は入力グラフの中からノードを無作為抽出し，そのノード属性で分散表現を初期化するものとする．

4.3 最大重みマッチング利得の最大化

図 1 に予測された未知ノード集合 $\hat{U}_{t+1}$ と真の未知ノード集合 U_{t+1} の対応関係と予測された未知ノードの分散表現を同時に学習する深層学習モデル Deep MatchMax の概要を示す．メインアーキテクチャは PointNet で採用されている T-Net と 7 層の Multilayer Perceptron（MLP）から構成される．前処理で初期化された d_0 次元のベクトル集合 $\hat{U}_{t+1}$ を入力とし，アーキテクチャ内部で $d_1 \to d_2 \to d_3 \to d_1 + d_3 \to d_4 \to d_5 \to d_6$ の次元数の変化を経て d_0 次元のベクトル集合 $\hat{U}_{t+1}$ を出力とする．各層の具体的な次元数については 5. 章で述べる．また，各階層の MLP の重みは全てのノードで共有されており，3 層目

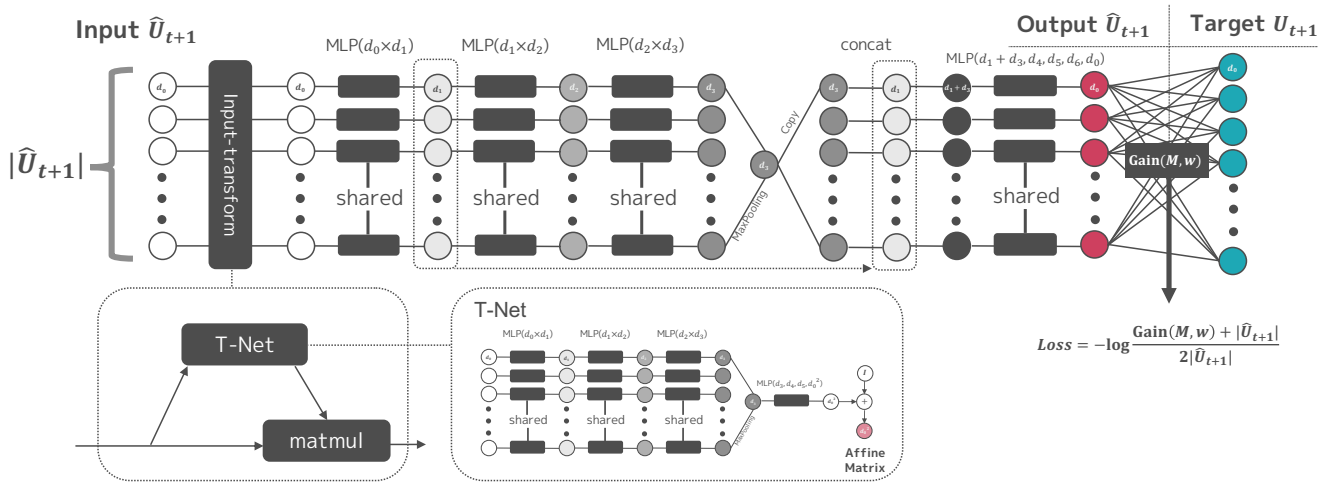


図 1　Deep MatchMax : Deep learning for matching gain maximization

の MLP の出力後に MaxPooling 関数を適用することで「順序についての不変性」の要求に応える．T-Net は入力される表現ベクトルの幾何学的変換への対応を目的とする Affine matrix の出力を行うアーキテクチャであり，PointNet で採用されている．Input-transformation の内部では T-Net で学習された Affine matrix $\in \mathbb{R}^{{d_0}^2 \times {d_0}^2}$ が全入力ベクトル $\mathrm{vec}(v) \in \mathbb{R}^{d_0}$ に積演算として適用され，各入力ベクトルに加わるノイズ（例えば，平行移動など）に対して頑健であるように学習される．真の未知ノード集合 U_{t+1} とその分散表現を教師として，式（3）を最適化するように訓練を行う．損失関数は以下の式で表される．

$$Loss = -\log \frac{\mathrm{Gain}(M,w) + |\hat{U}_{t+1}|}{2|\hat{U}_{t+1}|} \qquad (5)$$

$\mathrm{Gain}(M,w)$ は完全二部グラフ $K = ((\hat{U}_{t+1}, U_{t+1}), E)$，$E = \hat{U}_{t+1} \times U_{t+1}$ における最大重みマッチング利得である．また，

$$-1 \leqq \frac{\mathrm{Gain}(M,w)}{|\hat{U}_{t+1}|} \leqq 1$$

であるため損失関数は正規化された形を取っており，$\mathrm{Gain}(M,w)$ が大きいほど式（5）は 0 に近づく．

5. 評価実験

本章ではまず，人工データセットを用いてグラフ規模と最大重みマッチング利得の関係性の検証を行う．次に，提案手法の予測精度の評価を行う．

5.1 グラフ規模と最大重みマッチング利得の関係性

Deep MatchMax は二部グラフにおける最大重みマッチング利得を最大化するようにノード分散表現を学習する．以下では，Deep MatchMax で学習を行わなずに前処理で得られたノード分散表現の初期値をそのまま出力する手法をベースライン手法とする．本節では，ベースライン手法において二部グラフのノード数と最大重みマッチング利得がどのように関係するかを明らかにし，Deep MatchMax が有効に機能するグラフパターンを示す．

5.1.1 人工データセット

本実験では，簡単のため予測された未知ノード数 $|\hat{U}_{t+1}|$ と真の未知ノード数 $|U_{t+1}|$ は同じであるとし，M と置く．ノード分散表現は全ての要素が連続変数であるとし，正規分布に従うと仮定する．ノード分散表現の次元数は d とする．ノード分散表現の初期行列 $\hat{U}_{t+1}$ は下記の通りである．

$$\hat{\boldsymbol{U}}_{\boldsymbol{t+1}} = \begin{pmatrix} \mathcal{N}(\mu_0,\sigma_0),\ldots,\mathcal{N}(\mu_{d-1},\sigma_{d-1}) \\ \mathcal{N}(\mu_0,\sigma_0),\ldots,\mathcal{N}(\mu_{d-1},\sigma_{d-1}) \\ \vdots \\ \mathcal{N}(\mu_0,\sigma_0),\ldots,\mathcal{N}(\mu_{d-1},\sigma_{d-1}) \end{pmatrix} \in \mathbb{R}^{M \times d}$$

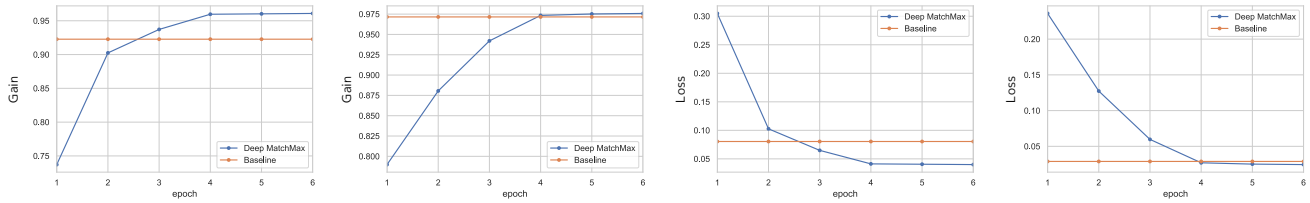
ただし，$\mathcal{N}$ は正規乱数であり，ノード間で seed は共有されており，seed もまた乱数で生成されるとする．実験では seed 中の μ は標準正規乱数，σ は一様乱数（0〜1）である．$\hat{U}_{t+1}$ と U_{t+1} は二部グラフを構成し，ノード分散表現同士のコサイン類似度をリンクの重みとする．$\hat{U}_{t+1}$ が最大重みマッチング利得の最大化を目的とする教師行列 U_{t+1} は下記の通りである．

$$\boldsymbol{U_{t+1}} = \begin{pmatrix} \mathcal{N}(\mu_0+\epsilon_0,\sigma_0+\hat{\epsilon}_0),\ldots,\mathcal{N}(\mu_{d-1}+\epsilon_{d-1},\sigma_{d-1}+\hat{\epsilon}_{d-1}) \\ \mathcal{N}(\mu_0+\epsilon_0,\sigma_0+\hat{\epsilon}_0),\ldots,\mathcal{N}(\mu_{d-1}+\epsilon_{d-1},\sigma_{d-1}+\hat{\epsilon}_{d-1}) \\ \vdots \\ \mathcal{N}(\mu_0+\epsilon_0,\sigma_0+\hat{\epsilon}_0),\ldots,\mathcal{N}(\mu_{d-1}+\epsilon_{d-1},\sigma_{d-1}+\hat{\epsilon}_{d-1}) \end{pmatrix} \in \mathbb{R}^{M \times d}$$

ϵ と $\hat{\epsilon}$ は誤差項であり，予測された未知ノードの統計量と真の未知ノードの統計量間の誤差を表現している．Deep MatchMax はこのギャップを埋めるように学習されることを期待される．誤差項は seed と同じくそれぞれ乱数生成され，ノード間で共有されるものとする．

5.1.2 ノード数と最大重みマッチング利得

ベースライン手法において，二部グラフのノード数が最大重みマッチング利得に及ぼす影響を考察する．ノード分散表現の次元数を $d = 10$ に固定し，ノード数 M を変化させた時の最大重みマッチング利得を図 3 に示す．横軸の M は対数軸であり，1 から 1000 までのノード数を表している．各ノード数のグラフごとに異なる seed と誤差項を持つ二部グラフを 100 回ずつ生成しており，縦軸はその最大重みマッチング利得の平均値を示す．黄色の曲線は教師行列 U_{t+1} に誤差項を持たせなかった場合（$\hat{U}_{t+1}$ と U_{t+1} は同じデータ分布），青色の曲線は教師行列 U_{t+1} が誤差項を含む場合を示している．誤差項を含む二部グラフ（青色）は誤差項を含まない二部グラフ（黄色）に比べて最



(a) Gain of test dataset ($M = 10$)　(b) Gain of test dataset ($M = 1000$)　(c) Loss of test dataset ($M = 10$)　(d) Loss of test dataset ($M = 1000$)

図 2　人工データセットにおける Deep MatchMax の学習過程

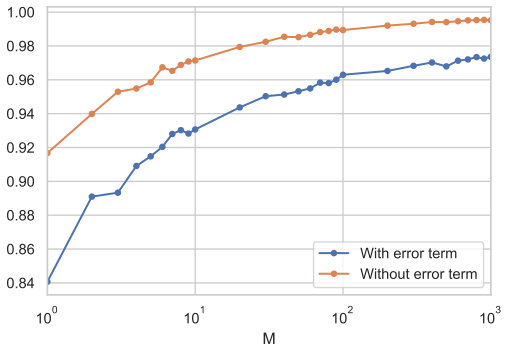


図 3　ノード数と最大重みマッチング利得

表 1　Deep MatchMax における各 MLP 層の次元数

param	d_0	d_1	d_2	d_3	d_4	d_5	d_6
次元数	10	64	128	1024	512	256	128

大重みマッチング利得が小さくなっている．これについては入力行列と教師行列のデータ分布が異なることが原因であり，誤差が大きければ大きいほど最大重みマッチング利得は小さくなると考えられる．一方で，誤差項を含む含まないに限らずノード数を増やすと最大重みマッチング利得が大きくなる傾向が見られる．これは膨大なノードが存在すれば，それだけ自身と類似するノードが存在する確率が上がるためであると考えられる．したがって，誤差項を含む二部グラフであったとしても，ノード数が多ければ最大重みマッチング利得は増大するため，改めてノード分散表現を学習し直すことで得られる利点は少なくなる．例えば，ノード数が 1000 の時，ベースライン手法の最大重みマッチング利得は 0.9735 を達成するため，ベースライン手法で十分な性能が担保される．以下では，Deep MatchMax の性能を評価するために，ノード数が少ない場合 ($M = 10$) と多い場合 ($M = 1000$) の 2 パターンで Deep MatchMax の学習を行い性能を比較する．

5.1.3　Deep MatchMax の学習

上記の人工データセット（$M = 10$，$M = 1000$，それぞれ 1000 のサンプル）を用いて Deep MatchMax の学習を行う．本実験の Deep MatchMax のモデルパラメータを表 1 に示す．

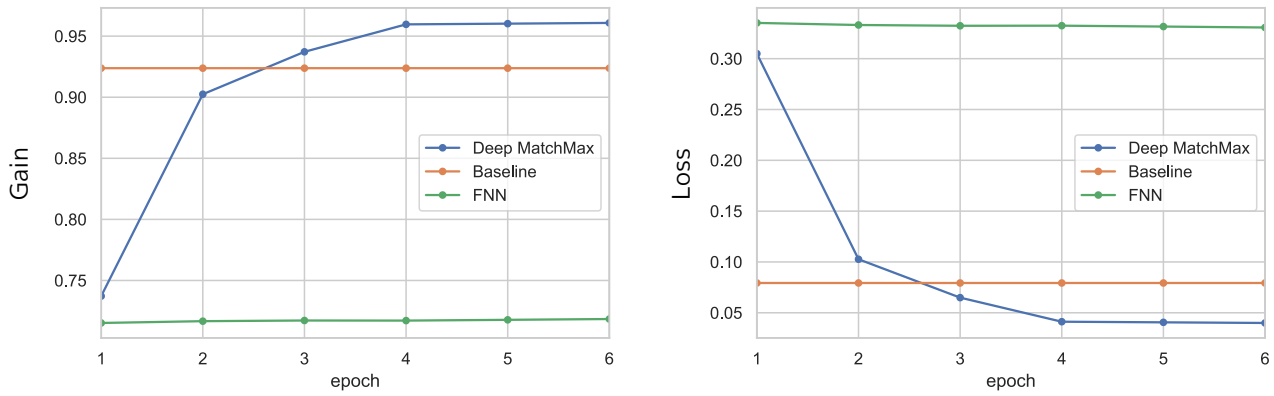
図 2（a）にグラフ規模が小さい場合の最大重みマッチング利得の推移を，図 2（c）に損失曲線を示す．横軸は epoch 数を表しており，青色が Deep MatchMax，黄色がベースラインの曲線を表す．これらの結果より，Deep MatchMax の学習結果がベースライン手法の最大マッチング利得をテストデータで大きく上回ることができていることが分かる．同様にして，グラフ規模が大きい場合の最大重みマッチング利得の推移を図 2（b）に，損失曲線を図 2（d）に示す．学習はグラフ規模が小さい場合と同様に学習は進むが，(a）と比較してベースライン手法の最大重みマッチング利得が大きいため Deep MatchMax で学習を行うことの利点は少ないこと分かる．また，モデルアーキテクチャの比較手法として Feed-Forward Neural Network（FNN）との学習過程の比較を行う．図 4（a）に最大マッチング利得の推移を，図 4（b）に損失曲線を示す．本タスクにおける損失関数では予測された未知ノード集合と真の未知ノード集合の最大重みマッチングを取得するが，入力順序が入れ替わってもこれらのノードペアは変わらないことを求められる．FNN はノード同士を接続する全結合層を有しており，入力順序が変わると出力結果も同時に変化する．そのため，図 4（b）の FNN のように損失関数は減少せず，学習が進まないため図 4（a）に示されるように最大重みマッチング利得も増大しない．したがって，Deep MatchMax が未知ノードの分散表現学習に有効なアーキテクチャであることが分かる．

6.　関 連 研 究

以下では関連研究として，ノードの分散表現学習手法として幅広く適用されているグラフ埋込手法や深層学習モデルを説明する．また，OWA を仮定しているタスクについて，Transductive 及び Inductive なタスクとの違いを示す．

6.1　グラフ埋込手法

ノードの分散表現を得る手法として様々なグラフ埋込手法が提案されている．グラフ埋込とは，グラフにおけるノード間の隣接関係に基づいてノードをベクトル空間に埋め込み，ノードの分散表現を得る手法である．ノードの分散表現を得ることで，ノード集合に対してベクトル空間に基づく機械学習手法を適用できるようになり，属性予測やリンク予測等の多様なタスクに応用可能となる．静的グラフに適用できる代表的なグラフ埋込手法としては，DeepWalk [8] が存在する．DeepWalk はグラフのランダムウォークにより得られるノードの系列に対して skip-gram [9] を適用することでノードの分散表現を得る手法である．現在はノード系列の作成方法を改良した node2vec [10] が広く知られている．また，属性付きの静的グラフに対するグラフ埋込では TADW [11] や DANE [12] が提案されている．これらの手法はノードの隣接性が高く，属性の類似度が高いノードは分散表現の類似度も高くなるように設計されている．動的



(a) Gain of test dataset ($M = 10$)　　(b) Loss of test dataset ($M = 10$)

図 4　人工データセットにおける Deep MatchMax および FNN の学習過程（$M = 10$）

グラフを対象としたグラフ埋込手法としては tNodeEmbed [13] や DDNE [14] が提案されている．これらの手法は Dynamic Network Link Prediction（DNLP）と呼ばれる,「ノード集合が固定されておりトポロジのみが時間変化する時系列グラフ」を対象としたリンク予測タスクにおいて広く用いられる手法である．ここで説明したグラフ埋込手法は全体の一部であるが, 筆者らの知る限り「ノード集合が時間変化する時系列グラフ」に適用可能なグラフ埋込手法や，トポロジの時間変化とノード属性を同時に考慮したグラフ埋込手法は存在せず, 全て CWA に基づいた手法となっている．

6.2　Graph Neural Networks

次に，ノードの分散表現を学習する深層学習モデルについて説明する．近年, グラフ構造に対して畳み込み演算を行う Graph Convolutional Networks（GCN）[15] とそれに関する研究が盛んに行われている．GCN はグラフフーリエ変換を用いる畳み込みのアプローチを取っており, 訓練パラメータがグラフラプラシアンの固有ベクトルに依存するという課題があった．そのため, 学習データに存在しないグラフ構造やノードについては学習済みモデルを適用できないため，Transductive なタスクにのみ応用可能であった．その後，訓練パラメータがグラフ構造に依存しないようにアテンションに基づく Graph Attention Networks（GAT）[16] やノードの近傍から特徴を集約する Aggregate 関数を学習する GraphSAGE [17] が提案された．これらのモデルは学習データに存在しないノードに対しても推論を行える Inductive なタスクにも適用可能な深層学習モデルである．Transductive 及び Inductive 双方のタスクは予め全てのノード集合とリンク集合が与えられているため CWA を仮定したタスクであると言える．また，動的グラフを対象とした深層学習モデルも存在し，STGCN [1] や DCRNN [2] が提案されている．しかし，これらのモデルはノード集合が時間変化する時系列グラフやトポロジの時間変化には対応しておらず，全て CWA を仮定した手法となっている．

先に示したように，ここまでで研究されている動的グラフを対象としたノードの分散表現学習手法は全て CWA に基づいている．OWA を仮定したグラフ予測に取り組むにはノード集合が時間変化する時系列グラフを対象とした未知ノード集合の分散表現学習手法が求められる．

7.　結　　論

本稿では，将来新たに出現する未知ノードの分散表現学習のための新たなフレームワーク Deep MatchMax を提案した．メインアーキテクチャとして PointNet を採用することで入力ノードの順序に影響されない学習を実現した．実験では，グラフ規模と最大重みマッチング利得の関係性を明らかにし，筆者らのモデルが有効に機能する状況を示した．実データに対して Deep MatchMax を適用し，性能評価を行う必要性が挙げられるがこれは今後の課題とする．

文　　献

[1] Yu Bing, Haoteng Yin, and Zhu Zhanxing. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. In *Proceedings of the IJCAI*, 2018.

[2] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *Proceedings of the ICLR*, 2018.

[3] Huiyuan Chen and Jing Li. Exploiting structural and temporal evolution in dynamic link prediction. In *Proceedings of the CIKM*, pp. 427–436, 2018.

[4] Raymond Reiter. On closed world data bases. In *Proceedings of the Logic and Data Bases*, pp. 55–76, 1977.

[5] Ismail Ilkan Ceylan, Adnan Darwiche, and Guy Van den Broeck. Open-world probabilistic databases. In *Proceedings of the KR*, pp. 339–348, 2016.

[6] Baoxu Shi and Tim Weninger. Open-world knowledge graph completion. In *Proceedings of the AAAI*, pp. 1957–1964, 2018.

[7] Charles Ruizhongtai Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the CVPR*, pp. 77–85, 2017.

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

[9] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *Proceedings of the ICLR*, 2013.

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

[11] Cheng Yang, Zhiyuan Liu, Deli Zhao, Maosong Sun, and Edward Y. Chang. Network representation learning with rich text information. In *Proceedings of the IJCAI*, pp. 2111–2117, 2015.

[12] Hongchang Gao and Heng Huang. Deep attributed network embedding. In *Proceedings of the IJCAI*, pp. 3364–3370, 2018.

[13] Uriel Singer, Ido Guy, and Kira Radinsky. Node embedding over temporal graphs. In *Proceedings of the IJCAI*, pp. 4605–4612, 2019.

[14] Taisong Li, Jiawei Zhang, Philip S. Yu, Yan Zhang, and Yonghong Yan. Deep dynamic network embedding for link prediction. *IEEE Access*, Vol. 6, pp. 29219–29230, 2018.

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

[16] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *Proceedings of the ICLR*, 2018.

[17] William L. Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the NIPS*, pp. 1024–1034, 2017.