

Weisfeiler-Lehman アルゴリズムに基づく 新しいグラフ構造間距離の提案

方 鐘熙^{1,a)} 黄 健明^{1,b)} 笠井 裕之^{1,2,c)}

概要: 有効なグラフ距離の定義は、距離の理論的妥当性、時間複雑性、及びグラフ間の距離としての有効性を考慮する必要があるため、グラフ機械学習における困難な課題である。本稿では、グラフカーネルでよく用いられる Weisfeiler-Lehman (WL) アルゴリズムの欠点に対処し、新しいグラフ構造間の距離を提案する。具体的には、まず WL アルゴリズムを構造解析の観点から分析し、カテゴリラベルの整合性のみに基づいてノードを識別することは、重要な構造情報を十分に捉えないことを論じる。そこで、カテゴリラベルを用いる代わりに、WL 部分木間のノード距離を木構造編集距離で定義し、複雑なグラフ構造を測定することを試みる。さらに、その計算のための効率的なアルゴリズムを提案する。最後に、提案したノード距離を応用し、最適輸送の枠組みを利用した埋め込み空間上のグラフ距離を定義する。要約すると、我々は2つのノード間の木構造編集距離を定義し、それをグラフレベルに反映させる。グラフ分類課題に対する数値実験の結果、提案するグラフ Wasserstein 距離は従来手法と同等以上の性能を持つことが示された。

A novel graph distance based on Weisfeiler-Lehman algorithm

1. はじめに

実世界の多くのデータセットは、グラフ構造として表現することができる。したがって、一般化可能なグラフ機械学習モデルは幅広い応用が期待される。このようなグラフ機械学習モデルを構築するには、グラフの属性情報とグラフに内在する構造情報を正確に捉える必要がある。グラフニューラルネットワーク (GNN) は近年、その優れた性能をもって、知識グラフや推薦システムにおいて目覚ましい成果を上げている。しかし、属性情報量に特化しているため、構造情報を記述することができない。DeepWalk [14] や Node2vec [8] などの確率的探索アルゴリズムを用いた他のグラフ埋め込み手法についても同様である。これらは友人・知人関係などのネットワーク情報の

抽出に成功しているが、確率的に訪問したノードの順次接続にしか対応しておらず、構造情報の把握には不十分である。さらに、これらの多くはノードの属性を利用することができず、埋め込み能力に難がある [5]。一方、グラフ編集距離 [7]、Gromov-Wasserstein 距離 [12] や Fused Gromov-Wasserstein 距離 [17] など、構造情報の違いを測定するために設計された手法もある。しかし、これらの手法は問題が NP 困難であるため、近似解法をもっても入力ノードに対して $O(n^3)$ 必要とし、実用的な応用は限定的である。また、現在の一般的なニューラルネットワークモデルでも、構造情報を正確に捉えることはできない。

そのため、本稿ではグラフ構造の比較に着目する。本研究では、Weisfeiler-Lehman (WL) カーネル [15] と Wasserstein WL (WWL) カーネル [16] を掘り下げる。そして、より効果的に構造情報を得る方法を提案し、グラフ構造間の新しい距離を定義する。グラフカーネルは、グラフ上の内積を計算するカーネル関数で、その多くは、部分グラフを比較してグラフの類似度を測る $\mathcal{R}$ -convolutional 理論 [9] に基づいて構築されている。グラフカーネルの文献では、鍵となる部分グラフを捉えるために、様々な種類のグラフ分解法が提案されている。その中でも、WL カーネルは、グラフ分類タスクにおいて高い精度を提供する最も信頼性

¹ 早稲田大学 基幹理工学研究科 情報理工・通信専攻
Department of Computer Science and Communications Engineering, FSE Graduate School, WASEDA University, 3-4-1 Okubo, Shinjuku-ku, Tokyo 169-8555, Japan

² 早稲田大学 基幹理工学部 情報通信学科
Department of Computer Science and Communications Engineering, FSE Graduate School, WASEDA University, 3-4-1 Okubo, Shinjuku-ku, Tokyo 169-8555, Japan

a) fzx@akane.waseda.jp

b) koukenmei@toki.waseda.jp

c) hiroyuki.kasai@waseda.jp

が高く一貫性のある手法の一つとして知られている。

WL アルゴリズムは、ノードの近傍構造情報をハッシュ値で圧縮することで、構造の類似性を表現している。しかし、構造情報をハッシュ化すると、異なる構造情報は異なる値に射影されるため、構造情報間の距離は1か0の値でしか評価できない。そこで、WL アルゴリズムの仕組みに従い、ノードの部分グラフ (WL 部分木) を構築し、ノードの周辺構造情報に対して適用する。WL 部分木間の距離を定義することで、WL アルゴリズムがノードラベルが同じかどうかだけを区別し、ノード間の有効な距離を定義できない問題を克服する。

ノード間の有効な距離を定義するために、我々のフレームワークはまず、各ノードに対して根付きの無秩序な木である WL 部分木を構築する。次に、ノード間距離を定義するために、木構造編集距離を使用する。無秩序木間の編集距離は MAX SNP 困難問題 ($P = NP$ でない限りどちらの問題も多項式時間近似方式を持たない) であることを考慮し、これを解くために近似解法を採用する。近似解法を使用する条件として、WL 部分木の全種類の完全部分木を列挙する必要がある。このために、Schwartz-Zippel の補題に基づき、WL 部分木の構築とともに全ての完全部分木の型を列挙する効率的なアルゴリズムを提案する。さらに、埋め込み空間上のノード間距離、すなわち WL 部分木間の木編集距離を最適輸送の基底距離とみなして、グラフ Wasserstein 距離を定義する。

2. 準備

本節では、本稿で扱うグラフと木に当たって必要な数学的定義について述べる。さらに、Weisfeiler-Lehman (WL) アルゴリズム、WL カーネル、及び Wasserstein 距離について必要な知識を紹介する。

太字の小文字 $\mathbf{x}$ と大文字 $\mathbf{X}$ は、それぞれベクトルと行列を表す。 $\mathbf{X}_{i,j}$ は $\mathbf{X}$ の (i,j) の要素を表す。 $\mathbb{R}_+^n$ は非負の n 次元ベクトル空間、 $\mathbb{R}_+^{m \times n}$ は非負の $m \times n$ サイズの行列空間を表す。 Δ_n は n 個のビンを持つ確率シンプレックスを表す。 δ_x は x でのディラック関数である。 $\mathbf{1}_n = (1, \dots, 1)^T \in \mathbb{R}^n$ である。 $\{\cdot\}$ は要素の重複を許さない集合を表す。一方、 $\{\cdot\}$ は要素の繰り返しを許容する多重集合を表す。 a_i からなる集合 $\mathcal{A}$ を、 $\mathcal{A} = \{a_1, a_2, \dots\}$ と表記する。 G は、ノードの集合 V とエッジの集合 $E \subseteq V^2$ を持つグラフ構造のデータで、 $G(V, E)$ と表記する。 G は無向グラフであり、 G のノード数は $|V|$ である。さらに、 G のノード v の近傍は $\mathcal{N}_G(v) = \{u \in V \mid (v, u) \in E\}$ と表記する。 $\deg(v)$ は v の次数を表す。 v には、カテゴリラベルと連続属性ベクトルがあり、それぞれ $\ell(v) \in \mathbb{N}$, $\mathbf{x}_v \in \mathbb{R}^d$ と表記する (d はノード特徴の次元である)。 T は木であり、ここでは特に根付きで順序のない WL 部分木を指す。根が v である木 T については、 $T(v)$ と表記する。根ではな

いノード v の親ノードは $\text{parent}(v)$ であり、 v は $\text{parent}(v)$ の子である。 T のノード、エッジと葉の集合をそれぞれ $\mathcal{V}(T), \mathcal{E}(T)$ と $\mathcal{L}(T)$ と表記する。 T の部分木 T' は、ノード $v \in \mathcal{V}(T)$ に対して、 $\text{parent}(v)$ が $v \in \mathcal{V}(T')$ を意味するとき完全であり、完全部分木 t と表記する。また、根が v である完全部分木を $t(v)$ と表記する。 T のノード v の深さを $\text{dep}_T(v)$ と表記する。 T_1 から T_2 への同型を $T_1 \approx T_2$ と表記する。これらの木 T に関する表記は完全部分木 t に対しても同様に使用する。

Weisfeiler-Lehman アルゴリズム.

グラフ同型性判定問題とは、2つの有限グラフが同型か否かを判定する計算問題で、NP-intermediate 問題 [1] として知られている。Weisfeiler-Lehman (WL) テストはこの問題の線形時間近似解であり、その有効性はほとんどのグラフで確認されている [2]。ノードとその近傍のラベルを集約して順序付き文字列を作成し、これらの文字列をハッシュ化して新しいノードラベルを作成することで実現される。反復回数が増加するにつれて、これらのラベルは各ノードのより大きな近傍を網羅するようになり、より拡張された部分構造を比較できるようになる。WL アルゴリズムは、各ノードラベルを複数回更新する再帰的方式に従う。グラフ $G(V, E)$ に対して、WL アルゴリズムの k 番目の反復における各ノード $v \in \mathcal{V}$ のノードラベルを $\ell^{(k)}(v)$ とする。特に、 $\ell^{(0)}(v)$ を本来グラフが持つカテゴリラベルとする。各ノードの更新式は、次式で与えられる。

$$\ell^{(k+1)}(v) = \text{HASH} \left(\ell^{(k)}(v), \{ \ell^{(k)}(u) \mid u \in \mathcal{N}_{G(v)} \} \right). \quad (1)$$

ノード v に対して WL アルゴリズムを h 回実行すると、元のノードラベルを含む $h+1$ 個の異なるノードラベルの列が得られる。これをノード v の WL 特徴量と呼ぶ。

Weisfeiler-Lehman カーネル.

Weisfeiler-Lehman カーネルは、2つのグラフの類似度を、グラフの特徴ベクトルの内積を用いて定義する。

$$k_{WL}(G, G') = \varphi(G, \Sigma_h)^T \varphi(G', \Sigma_h), \quad (2)$$

ここで、 $\varphi(\cdot, \cdot)$ は対応するグラフのグラフ特徴ベクトル、 $\Sigma_h = \{\ell_0, \ell_1, \dots\}$ は WL アルゴリズムの h 回の繰り返しで G と G' に現れる全種類のノードラベルの集合である。さらに、 $\varphi(G, \Sigma_h)$ は、 $(\text{cnt}(G, \ell_0), \text{cnt}(G, \ell_1), \dots, \text{cnt}(G, \ell_i))$ と定義される。ここで、 $\text{cnt}(\cdot, \cdot)$ は G 内の ℓ_i の出現回数を返す関数である。

Wasserstein 距離.

Wasserstein 距離は、2つの確率分布間で最適な輸送計画を見つけることで輸送コストの最小値を計算する最適輸送問題 (Optimal Transport (OT)) に由来する。元の Monge の問題は条件が厳しく解くのが難しいため、既存の OT は通常 Monge-Kantorovich 問題を指す。離散の OT は以下

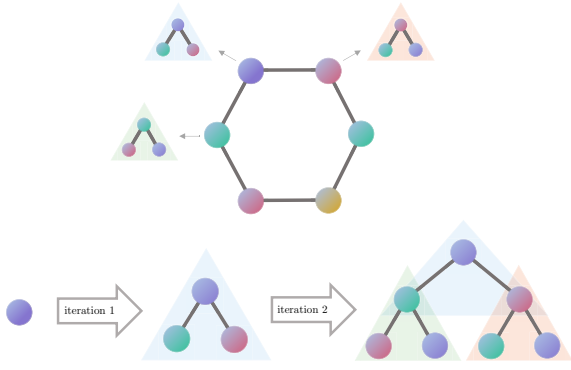


図 1 WL 部分木の構築図.

のように定義される. $\Delta_m = \{\mathbf{a} \in \mathbb{R}_+^m \mid \sum_{i=1}^m \mathbf{a}_i = 1\}$ と $\Delta_n = \{\mathbf{b} \in \mathbb{R}_+^n \mid \sum_{j=1}^n \mathbf{b}_j = 1\}$ は同じ行列空間上の m と n のヒストグラムのシンプレックスである. その確率測度はそれぞれ $\alpha = \sum_{i=1}^m \mathbf{a}_i \delta_{x_i}$ と $\beta = \sum_{j=1}^n \mathbf{b}_j \delta_{y_j}$ で表記される. $\mathbf{C} \in \mathbb{R}_+^{m \times n}$ は距離行列であり, 特に, $\mathbf{C}_{i,j}$ は行列空間の i と j 地点の間の輸送コストを示す. $\mathbf{C}_{i,j}$ が厳密な距離であるとき, α と β 間の全体の輸送コストは p -Wasserstein 距離として定義される.

$$\mathcal{W}_p(\alpha, \beta; \mathbf{C}) = \left(\min_{\mathbf{P} \in \mathbf{U}(\mathbf{a}, \mathbf{b})} \sum_{i=1}^m \sum_{j=1}^n (\mathbf{C}_{i,j})^p \mathbf{P}_{i,j} \right)^{\frac{1}{p}}, \quad (3)$$

ここで, $p \in [1, \infty)$, $\mathbf{U}(\mathbf{a}, \mathbf{b}) = \{\mathbf{P} \in \mathbb{R}^{m \times n} \mid \mathbf{P}\mathbf{1}_n = \mathbf{a}, \mathbf{P}^T \mathbf{1}_m = \mathbf{b}\}$, $\mathbf{P}$ は輸送計画である輸送行列を表す. 一般に, Wasserstein 距離の計算量は $n = m$ のとき, $\mathcal{O}(n^3 \log(n))$ である. また, [4] が近似解法である Sinkhorn's algorithm を提案した. これにより, 計算量が $\mathcal{O}(n^2/\varepsilon^2)$ に削減された. また, ε が 0 に近いとき, Sinkhorn 距離は元の解に近くなり, ε が 0 のとき, 両者は一致する. さらに, Sinkhorn アルゴリズムは GPU で計算できるため, さらに計算速度を高速化を可能とする.

3. グラフ間の Wasserstein Weisfeiler-Lehman 距離

Weisfeiler-Lehman (WL) アルゴリズムは, 式 (1) の再帰的なラベル更新を行う. 構造解析では, WL アルゴリズムで得られた新しいラベルは, WL 部分木と呼ばれる新しく構築された木に対応する整数のハッシュ値と見なすことができる. WL 部分木は根付きの無秩序木であり, 以下の特徴を持つ: 木の根は WL アルゴリズムのターゲットノードである; 均衡の取れた木構造である; 木の高さは常に反復回数に等しい; 高さ k に現れるノードは常に高さ $k+2$ に現れる. 図 1 は各反復における構築過程の説明図である. 重要な点は, WL 部分木には, 対象ノードとその近傍ノードとの構造情報であるノード間接続情報が含まれていることである. あるノードに対して WL アルゴリズムを h 回繰り返すことで, WL 部分木はそのノードから h ホップ

以内の部分グラフの接続情報を得ることができる. h が十分に大きければ, WL 部分木はグラフ全体の構造情報を含む. WL アルゴリズムは, この重要な構造情報を整数値で表現されるため失われる.

3.1 WL カーネルと WWL カーネルの測定力の限界

WL カーネルはグラフの分類タスクで成功を収めているが, 式 (2) の測定手法が単純なため, 複雑なグラフ構造間の比較には限界がある. また, 式 (2) は $k_{WL}(G, G') = \sum_{i=0}^h \sum_{v \in \mathcal{V}} \sum_{v' \in \mathcal{V}'} \delta(\ell^{(i)}(v), \ell^{(i)}(v'))$ に書き換えることができる. ここで, $\delta(\cdot, \cdot)$ は引数が異なるととき 0, 等しいとき 1 を返す関数である. この式は, WL カーネルが 2 つのノード間の類似度を 1 か 0 の値で評価すること示している. 前節で説明したように, 各ノードラベルは WL 部分木に対応しており, ディラック関数では貴重な構造情報を失うことになる. そのため, WL カーネルのグラフレベル, ノードレベルでの類似度測定は, 複雑なグラフ構造を測定するのに不十分である. WWL では, グラフレベルでの計測能力を向上させるために, より高度な計量として Wasserstein 距離を採用している. しかし, WWL のカテゴリ埋め込みの場合, ノードレベルでの測定に問題が残る. WWL では, WL 特徴をノード特徴として扱い, 正規化ハミング距離を用いて基底距離を定義している. WL アルゴリズムの特性として, 反復回数 k_0 ($k_0 \geq 0$) 回目で二つのラベルが異なる場合, その後の反復回数 $k' > k_0$ での更新によって得られるそれらのラベルもまた異なる. これは, 2 つの WL 特徴間のハミング距離は, h 回の繰り返して最大 $h+1$ 個の異なる値しかとれないことを意味する. この状況は, 開始ラベルが異なるが近傍が類似しているノード間の類似性を捉えるには粗すぎるという問題を引き起こしてしまう. これらの問題とは別に, 最適輸送の実用的な効果は基底距離に依存し, 2 つのグラフのペアワイズマッチングが悪くなる可能性がある.

3.2 WL 部分木間の木構造編集距離

ノード間の距離を計算するために WL 特徴を用いるのではなく, WL 部分木に着目して木構造間の距離を計算する. 具体的には, ノード v と v' の WL 部分木間の距離を, 木構造編集距離を用いて定義する. 無秩序木間の木構造編集距離は MAX SNP 困難な問題であり, 木構造編集距離を用いてグラフ間のペアワイズ距離を計算することは困難である. そこで, 順序付き編集距離を L^1 ノルム空間に埋め込む近似解法を使用する [6].

提案するアルゴリズムを正式に紹介する前に, 図 2 を用いて必要ないくつかの表記を説明する. 図 1 の青ノードが v_1 であると仮定する. WL アルゴリズムを 2 回繰り返すことで, v_1 をルートとする WL 部分木が得られ, これを $T(v_1)$ とする. $T(v_1)$ のノードは $v_1, \dots, v_7$ の 7 つである.

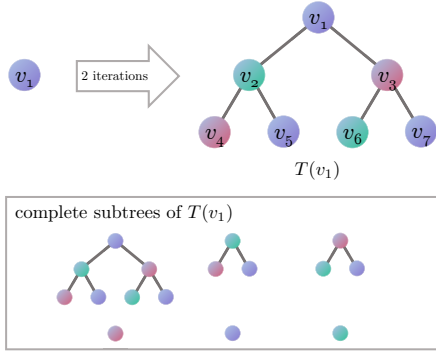


図 2 木 $T(v_1)$ の完全部分木.

v_1, v_5 と v_7 は本来同じだが, $T(v_1)$ では全てのノードを異なるものとして扱い, $T(v_1)$ のノード集合を $\mathcal{V}(T(v_1))$ と表記する. $T(v_1)$ の完全部分木は $t(v_1), t(v_2), t(v_3), t(v_4), t(v_5), t(v_6), t(v_7)$ の 7 つである. $t(v_5)$ は $t(v_7)$ に近いので, 同一と見なします. したがって, 完全部分木は 6 種類となる.

WL 部分木間の木構造編集距離.

まず, v と v' の WL 部分木を構成し, $T(v)$ と $T(v')$ の特徴ベクトルの間の L^1 ノルムを距離関数 $d_\phi: \mathcal{T} \times \mathcal{T} \rightarrow \mathbb{N}$ を用いて計算する. ここで, $\mathcal{T}$ は WL 部分木の集合である. 距離関数は次のように定義される.

$$d_\phi(T(v), T(v')) = \|\phi(T(v), \mathcal{U}) - \phi(T(v'), \mathcal{U})\|_1, \quad (4)$$

ここで, $\phi(\cdot, \cdot)$ は対応する WL サブツリーの特徴ベクトル, $\mathcal{U} = \{t_1, t_2, \dots\}$ は $T(v)$ と $T(v')$ の全種類の完全部分木からなる集合であり, $\mathcal{U}$ 内の任意の二つの完全部分木 t_i, t_j は $i \neq j$ に対して $t_i \not\cong t_j$ である. また, $\phi(T, \mathcal{U})$ を $(cnt(T, t_1), \dots, cnt(T, t_{|\mathcal{U}|})) \in \mathbb{N}^{|\mathcal{U}|}$ と定義する. ここで, $cnt(T, t_i)$ は t_i が T 中に現れる回数を返す関数である.

$\phi(\cdot, \cdot)$ を効率よく計算するアルゴリズム.

$\phi(T(v), \mathcal{U})$ と $\phi(T(v'), \mathcal{U})$ を計算するためには, まず $T(v)$ と $T(v')$ に現れる全種類の完全部分木を知る必要がある. 完全部分木の種類をすべて洗い出し, それぞれを WL 部分木で探索するのは複雑すぎるため, 効率的なアルゴリズムを提案する. その方法は, Schwartz-Zippel の補題を用いて, post-order 深さ優先探索 (DFS) でノードを訪問する際に各完全部分木を整数値にマップするハッシュ関数を設計することである. Schwartz-Zippel の補題は, その仮定のもとで 2 つの多変数多項式が等しい確率を与えることができる. 我々の提案手法は, 各完全部分木にその多変数多項式を持たせることで, ハッシュ衝突の確率を証明できるようにすることである. コンピュータは変数を直接表現することができないので, 変数を乱数に置き換える. 高さ h の WL 部分木 T が与えられたとき, T の深さ h 以外の深さにそれぞれ 1 つの乱数を与える: 深さ i に対して異なる正の乱数 $r_{i \neq h} \in (0, M]$, $i \in \{0, 1, \dots, h-1\}$ (ただし M は最大整数値) を設定し, 深さ h に対して $r_h = 0$ を設定する. 表記を簡潔にするため, $p(v)$ を $t(v)$ の多項式,

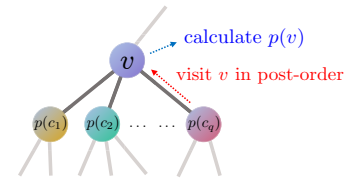


図 3 式 (5) の説明図 (1).

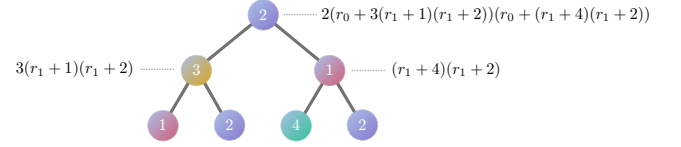


図 4 式 (5) の説明図 (2).

c_i を v の i 番目の子とする. 多項式を動的に計算するために, 我々は T を post-order DFS で探索し, v を訪問する際に, v の子を既に訪問していることを保証する. 図 3 に示すように, $\{p(c_i)\} \in \Sigma_0$ を計算後, v を訪れたとき関数 HASH_{sz} を使って $p(v)$ を計算する. HASH_{sz} は次式で定義される.

$$\begin{aligned} p(v) &= \text{HASH}_{sz} \left(\ell^{(0)}(v), r_{\text{dep}_t(v)}, \{p(c_i)\} \right) \\ &= \ell^{(0)}(v) \times (r_{\text{dep}_t(v)} + p(c_1)) \times \\ &\quad \cdots \times (r_{\text{dep}_t(v)} + p(c_q)), \end{aligned} \quad (5)$$

ここで, q は v の子の数である. 図 4 は, アルゴリズムを直感的に説明する図である. $v \notin \text{leaf}(T)$ の $t(v)$ に対応する多項式は $\{r_{\text{dep}_t(v')}\}_{v' \in \mathcal{V}(t(v)) \setminus \mathcal{L}(t(v))}$ 個の変数を持ち, 次数は $|\mathcal{L}(t(v))|$ となる. 次の定理は, Schwartz-Zippel の補題が $\text{HASH}_{sz}(\cdot, \cdot, \cdot)$ の衝突確率の上界を与えることができることを示している.

定理 1. $p(v_1)$ と $p(v_2)$ をはそれぞれ完全部分木 $t(v_1)$ と $t(v_2)$ に対応する多項式であると仮定する. このとき, $p(v_1)$ と $p(v_2)$ の衝突確率は $\frac{|\mathcal{L}(t(v_1))| + |\mathcal{L}(t(v_2))|}{M}$ で上から抑えられる.

Proof. $p(v_1)$ と $p(v_2)$ はそれぞれ 2 つの完全部分木 $t(v_1)$ と $t(v_2)$ に対応する多項式とする. $p(v_1) - p(v_2) = P(\mathcal{R})$ は, F 上の次数 d の多項式で, $\mathcal{R} \subseteq \{r_i \mid i \in \{1, \dots, h-1\}\}$ であると仮定する. また, $d \leq |\mathcal{L}(t(v_1))| + |\mathcal{L}(t(v_2))|$ である. S を F の有限部分集合とする. $r_i \in \mathbb{N}$, $r_i \in (0, M]$ と制約するので, $|S| = M$ となる. また, $\mathcal{R}$ は S からランダムに選択される. Schwartz-Zippel の補題により, $\Pr[P(\mathcal{R}) = 0] \leq \frac{d}{M} \leq \frac{|\mathcal{L}(t(v_1))| + |\mathcal{L}(t(v_2))|}{M}$ となる. $\square$

定理 1 により, 十分に大きな M を選択することで, 衝突確率を十分に低い数値に抑えることができる. したがって, 各多項式 $p(v)$ は通常, 完全なサブツリー $t(v)$ のハッシュ値であることがわかる. 式 (5) はハッシュ関数であると言える.

計算量解析.

提案するアルゴリズムの計算量解析を行う. これまでの説明では WL 部分木が構築されていることを前提としていたため, ここでは, まず WL 部分木の構築方法を説明する. 式 (5) は DFS 内に実装されているので, WL 部分木の構築も DFS を使用する. この方法の利点は, WL 部分木を構築しながらハッシュ値を計算できることである. つまり, 1つの DFS の枠組みの中で, アルゴリズムを完結させることができる. この手法では各ノードの近傍ノードを記録しておくだけでよい. ノード v を基点とした DFS は, まず v を訪問し, 次に v のまだ訪問していない全ての隣接ノード (この場合, グラフ上の v の隣接ノード) の中から一つを取り, その隣接ノードを基点に再帰的に DFS を行う. 木構造の計算量は, DFS の時間計算量と同じであり, $O(|V(T)| + |E(T)|)$ となる. 式 (5) の計算量が $O(1)$ であるとする, 全部で $|V(T)|/\mathcal{L}(T)|$ 回実行するため, 総計算量は $O(|V(T)| + |E(T)| + |V(T)|/\mathcal{L}(T)|) < 2O(|V(T)| + |E(T)|)$ となる. これは, 構築する WL 部分木に対して線形複雑度である.

グラフ Wasserstein 距離.

この小節では, 新しいグラフ距離について述べる. 我々の提案するグラフ距離は木構造編集距離を Wasserstein 距離と組み合わせている. 木構造編集距離をグラフレベルで反映させることにより, より複雑なグラフ構造を測定することを目指す.

提案するグラフ Wasserstein 距離は, あるグラフのノードと別のグラフのノードを一致させるための差を測定するもので, グラフアライメント問題として定式化される. これは, 離散 Monge 問題に類似している. そこで, グラフアライメント問題を OT 問題に変換し, グラフ間の輸送コストをグラフ距離として定義する. 2つのグラフ G と G' を仮定する. G と G' の各ノードの $\phi(\cdot, \cdot)$ を計算し, 同じ計量空間に埋め込む. 2つのグラフのノードはそれぞれ $x_1, \dots, x_{|V|}$ と $y_1, \dots, y_{|V'|}$ に位置する. ここでは, G のノードを始点, G' のノードを終点と考える. このとき, $\mathbf{a}$ と $\mathbf{b}$ の2つのヒストグラムは, それぞれ確率シンプレックス $\Delta_{|V|}$ と $\Delta_{|V'|}$ で定義される. さらに, 位置 $x_1, \dots, x_{|V|}$ 上に重み $\mathbf{a}$ を持つ離散測度 $\mu(G)$ として, $\mu(G) = \sum_{i=1}^{|V|} \mathbf{a}_i \delta_{x_i}$ を定義する. $\mu(G)$ は埋め込み空間上のノードの分布を表す. 同様に, 測度 $\mu(G') = \sum_{j=1}^{|V'|} \mathbf{b}_j \delta_{y_j}$ を定義する. 始点と終点に位置するノード間のペアワイズ距離を計算することにより, $\mathbf{C} \in \mathbb{R}^{|V| \times |V'|}$ を得る. 最後に, 式 (3) の $\mathbf{C}_{i,j}$ に $d_\phi(\cdot, \cdot)$ を入力し, 式 (6) を解けば $\mathbf{P}^*$ が求まる.

$$\mathbf{P}^* = \operatorname{argmin}_{\mathbf{P} \in \mathbf{U}(\mathbf{a}, \mathbf{b})} \sum_{i=1}^{|V|} \sum_{j=1}^{|V'|} d_\phi(T(v_i), T(v_j)) \mathbf{P}_{i,j}. \quad (6)$$

$d_{i,j}$ を距離 $d(\cdot, \cdot)$ で計算されたコスト行列 $\mathbf{C}$ とする. 提案

するグラフ Wasserstein 距離は $\mathcal{W}_1(\mu(G_1), \mu(G_2); (d_{\phi_{i,j}}))$ として導出することができる. OT の重要な性質は, 基底距離が距離の公理を満たすとき, 2つの確率の間に厳密な距離を与えることである. したがって, 提案する距離は厳密な距離関数である.

4. 実験

提案するグラフの距離関数に対して評価を行う. 評価指標として, グラフカーネルによるグラフ分類の実験を用いる. 比較対象は代表的なグラフカーネルを用いる: Shortest-path カーネル (SP) [3], Fast Random Walk カーネル (FRM) [18], WL カーネル (WL) とその拡張手法である WL-OA [10], WWL. 使用するデータセットは最も使用される TUD データセット [13] である.

本実験では, グラフカーネルによるグラフの分類実験を行う. そのため, 次式を用いて提案手法をグラフの距離関数から類似度へ変換する.

$$k(G, G') = \exp\left(-\gamma \mathcal{W}_1\left(\mu(G), \mu(G'); (d_{\phi_{i,j}})\right)\right),$$

ここで, γ はパラメータである. 提案するグラフカーネルは半正定値であることが保証できないため, Krein Support Vector Machine (SVM) [11] を分類器として使用する.

10-fold 10 cross validation を用いて, 各手法グラフカーネルを 100 回実行し, その平均を結果として使用する. 結果を表 1 と表 2 に示す. 一番良い結果は太文字で示す.

表 1 グラフ分類精度 (1)

METHOD	MUTAG	COX2	BZR
SP	83.01±7.8	77.53±5.2	83.13±0.3
FRW	74.68±9.1	79.64±2.4	78.77±1.0
WL	82.30±7.6	79.77±5.1	85.71±4.3
WL-OA	82.72±7.1	80.29±3.5	87.42±4.3
WWL	84.80±8.1	79.68±4.1	87.16±3.5
Ours	87.66±6.9	81.62±4.8	88.00±3.6

表 2 グラフ分類精度 (2)

METHOD	ENZYMES	PROTEINS	IMDB-B
SP	42.69±1.0	76.02±3.8	57.92±5.1
FRW	28.65±5.2	66.60±2.9	70.88±4.4
WL	42.20±5.5	72.30±3.3	73.04±4.2
WL-OA	57.50±4.2	73.50±3.7	72.90±3.9
WWL	50.47±6.1	74.10±3.8	72.93±4.1
Ours	56.83±5.7	74.94±3.9	73.74±3.9

5. まとめ

ENZYMES と PROTEINS を除くデータセットにおいて、提案手法は従来手法より高い結果を得ることができた。また、いずれのデータセットにおいて、提案手法は WL と WWL より高い結果を得ることができた。総じて、提案手法は従来手法と同等またはそれ以上の性能を有することが示された。

参考文献

- [1] Babai, L.: Graph isomorphism in quasipolynomial time, *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing (STOC)*, pp. 684–697 (2016).
- [2] Balcilar, M., Héroux, P., Gaüzère, B., Vasseur, P., Adam, S. and Honeine, P.: Breaking the Limits of Message Passing Graph Neural Networks, *Proceedings of the 38th International Conference on Machine Learning (ICML)* (2021).
- [3] Borgwardt, K. M. and Kriegel, H.-P.: Shortest-path kernels on graphs, *IEEE international conference on data mining (ICDM)*, pp. 8–pp (2005).
- [4] Cuturi, M.: Sinkhorn distances: Lightspeed computation of optimal transport, *Advances in neural information processing systems (NeurIPS)*, Vol. 26, pp. 2292–2300 (2013).
- [5] Deng, C., Zhao, Z., Wang, Y., Zhang, Z. and Feng, Z.: GraphZoom: A Multi-level Spectral Approach for Accurate and Scalable Graph Embedding, *International Conference on Learning Representations (ICLR)* (2020).
- [6] Fukagawa, D., Akutsu, T. and Takasu, A.: Constant factor approximation of edit distance of bounded height unordered trees, *International Symposium on String Processing and Information Retrieval (SPIRE)*, Springer, pp. 7–17 (2009).
- [7] Gao, X., Xiao, B., Tao, D. and Li, X.: A survey of graph edit distance, *Pattern Analysis and applications*, Vol. 13, No. 1, pp. 113–129 (2010).
- [8] Grover, A. and Leskovec, J.: node2vec: Scalable feature learning for networks, *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining (ICKD)*, pp. 855–864 (2016).
- [9] Haussler, D.: Convolution kernels on discrete structures, Technical report, Technical report, Department of Computer Science, University of California at Santa Cruz (1999).
- [10] Kriege, N. M., Giscard, P.-L. and Wilson, R.: On valid optimal assignment kernels and applications to graph classification, *Advances in Neural Information Processing Systems (NeurIPS)*, Vol. 29, pp. 1623–1631 (2016).
- [11] Loosli, G., Canu, S. and Ong, C. S.: Learning SVM in Krein spaces, *IEEE transactions on pattern analysis and machine intelligence*, Vol. 38, No. 6, pp. 1204–1216 (2015).
- [12] Memoli, F.: On the use of Gromov-Hausdorff Distances for Shape Comparison, *Eurographics Symposium on Point-Based Graphics* (Botsch, M., Pajarola, R., Chen, B. and Zwicker, M., eds.), The Eurographics Association (2007).
- [13] Morris, C., Kriege, N. M., Bause, F., Kersting, K., Mutzel, P. and Neumann, M.: TUDataset: A collection of benchmark datasets for learning with graphs, *ICML 2020 Workshop on Graph Representation Learning and Beyond (GRL+)* (2020).
- [14] Perozzi, B., Al-Rfou, R. and Skiena, S.: Deepwalk: On-line learning of social representations, *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining (ICKD)*, pp. 701–710 (2014).
- [15] Shervashidze, N., Schweitzer, P., Van Leeuwen, E. J., Mehlhorn, K. and Borgwardt, K. M.: Weisfeiler-Lehman graph kernels., *Journal of Machine Learning Research (JMLR)*, Vol. 12, No. 9 (2011).
- [16] Togninalli, M., Ghisu, E., Llinares-López, F., Rieck, B. and Borgwardt, K.: Wasserstein Weisfeiler-Lehman Graph Kernels, *Advances in Neural Information Processing Systems (NeurIPS)*, Curran Associates, Inc., pp. 6436–6446 (2019).
- [17] Vayer, T., Chapel, L., Flamary, R., Tavenard, R. and Courty, N.: Fused Gromov-Wasserstein distance for structured objects, *Algorithms*, Vol. 13, No. 9, p. 212 (2020).
- [18] Vishwanathan, S. V. N., Schraudolph, N. N., Kondor, R. and Borgwardt, K. M.: Graph kernels, *Journal of Machine Learning Research (JMLR)*, Vol. 11, pp. 1201–1242 (2010).