

郵便番号による地点間の意味的・物理的距離の算出方式の提案

山本 陸矢[†] 王 元元^{††} 角谷 和俊^{†††} 河合由起子^{†,†††}

[†] 京都産業大学コンピュータ理工学部 〒 603-8555 京都府京都市北区上賀茂本山

^{††} 山口大学大学院創成科学研究科 〒 755-8611 山口県宇部市常盤台 2-16-1

^{†††} 大阪大学サイバーメディアセンター 〒 567-0047 大阪府茨木市美穂ヶ丘 5 番 1 号

^{†††} 関西学院大学総合政策学部 〒 669-1337 兵庫県三田市学園 2 丁目 1

E-mail: †{g1745310, kawai}@cc.kyoto-su.ac.jp ††y.wang@yamaguchi-u.ac.jp

あらまし MaaS (Mobility as a Service) が世界的に整備普及され始め、電車やバスなどの交通手段と ICT を組み合わせた移動支援サービスの研究開発が活発化している。移動支援サービスではユーザ行動分析に基づき情報推薦するが、近年ユーザの行動履歴の学習をサーバで実行せず、ユーザの端末で学習するフェデレーティッドラーニングが注目されている。しかしながら、従来の機械学習では、意味的近似や最適化の前処理としてテキストデータから One-Hot Encoding 等でベクトルを生成し特徴量を抽出するが、移動体端末では負荷が課題となる。具体的には、ユーザが訪問した地物の住所情報を用いた特徴量抽出では、One-Hot Encoding は単語出現数を次元とするベクトルを生成するためメモリ使用量を増加させ、計算コストが高くなる。そこで、本研究では、住所情報として郵便番号と番地へ変換することでベクトル距離計算のコストを削減しつつ、地点間の意味的類似距離ならびに物理的距離の同時算出手法を提案する。本稿では、米国の郵便番号を用いた意味的・物理的距離の算出を行い、従来手法の住所による意味的類似距離算出手法との比較実験、および緯度経度を用いた物理的距離算出手法との比較実験を行い、提案手法の有効性を検証する。

キーワード One-Hot Encoding, 意味的類似距離, 地理データ分析, 情報推薦・検索

1. はじめに

近年、MaaS の基盤整備が多くの都市で進められており、様々な交通手段を最適化し、シームレスに結び付けて効率良く移動できるサービスの開発が活発化している。これに伴い、電車やバス、レンタカーなどの交通手段が各都市で拡充され始めたことにより移動が便利になった。これに伴い、移動支援サービスの需要が増大しており、ユーザ行動分析に基づく情報推薦では、企業が大量のユーザの行動履歴を収集し、サーバ上で機械学習を実行させていたが、フェデレーティッドラーニング^(注1)と呼ばれるユーザの端末（スマホ）上で機械学習を実行させることで計算コストを削減しつつ、優れたモデル生成を実現可能とする新たな方式が注目されている。このような背景のもと、サービス提供者は推薦する地物（ホテルやスポット等）に対してユーザの嗜好性が高く、ユーザの移動地点間の距離が短い地物を検索、推薦することで満足度を高めることが重要となり、ユーザの嗜好と地物属性との類似距離や移動する地点間の物理的距離を機械学習から算出することが必要となる。例えば、ホテルのレビュー情報との類似距離算出やツイッターの内容に記載された位置情報を用いた地域性分析や距離算出に用いられている[1][2][3][4]。

機械学習では、テキストに対する前処理として単語の出現を 0/1 で表現できる One-Hot Encoding によりベクトルを生成す

るが、単語によるカテゴリ分類が困難なセンテンスに対してはメモリ使用量およびベクトル距離計算コストが高く、数万以上のレコードに対する学習時間は課題となっている。そのため、一般的にはレビューやツイートのセンテンスに対しては、自然言語分析により上位 300 単語程度を抽出し、さらにそれらから 2 衔程度のカテゴリに分類し、機械学習を実施する。近年ではセンテンスとして住所情報が空間特徴量分析に用いられており、京都市北区と大阪市北区の特徴ベクトルによる類似性算出に利用されたり、住所から緯度経度に変換し物理的距離を算出することで特徴ベクトルの要素として利用されている[5][6][7][8]。

本研究では、住所情報の空間特徴量分析として、空間的配置により付定されている郵便番号に着目し、住所情報の郵便番号変換による地点間のベクトル空間距離算出手法を提案する。提案手法は、住所から郵便番号に変換し、さらに番地、フロア情報、部屋番号の数値のみを抽出し、高々数次元のベクトルに変換する。これにより、地点間の意味的類似距離算出および物理的距離の算出時間およびメモリ使用量が大幅に削減でき、ユーザの行動履歴から POI の推薦サービス等のリアルタイム処理へ応用でき、移動情報等を取り扱うサービスの満足度の向上が期待できる。

本論文では、地物の住所から郵便番号と番地によるベクトルを生成し、地点間の意味的・物理的距離の算出手法を提案し、米国の 6 都市を対象に、従来手法の One-Hot Encoding と提案手法とのコサイン類似度による意味的距離の誤差の比較検証ならびに緯度経度と提案手法との物理的距離の誤差の比較検証を行ふ。また、従来手法と提案手法による 2 地点間の意味的距離

(注1) : <https://developers-jp.googleblog.com/2017/05/federated-learning-collaborative.html>

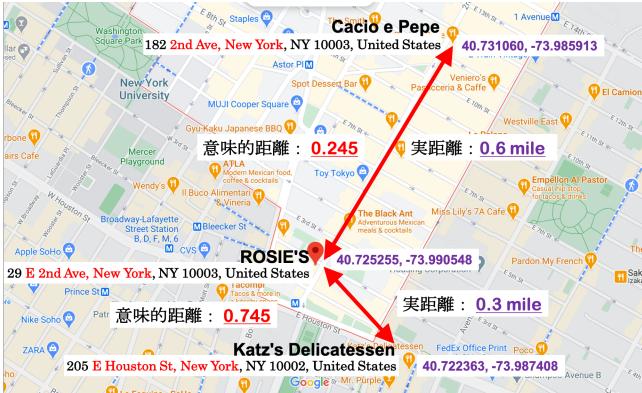


図 1 意味的距離・物理的距離の概要図

計算の処理時間およびメモリ使用量の比較検証を行う。

以下、2章では関連研究との比較、相違点について述べ、3章では郵便番号による地点間の意味的・物理的距離の算出方式について述べる。4章では、意味的距離ならびに物理的距離算出精度とメモリ使用量・処理時間を検証し、最後に5章でまとめと今後の予定について述べる。

2. 意味的・物理的距離による空間分析

2.1 意味的距離と物理的距離

本研究では、住所情報の空間特微量分析として、空間的配置により付定されている郵便番号に着目し、住所情報の郵便番号変換による地点間のベクトル空間距離算出方式を提案する。

意味的距離とはユーザの行動履歴からPOIの名称や住所などの位置情報の特徴を利用して算出される地点間の特徴ベクトルによる類似距離のことである。また、物理的距離は緯度経度を利用して算出される地点間の距離のことである。図1は3つの地点間の意味的・物理的距離が表示されており、例えばCacio e PepeとROSIE'Sの実距離0.6mileが物理的距離にあたり、意味的距離0.245は両方の住所情報の意味的な距離を示している。

2.2 関連研究

本章では空間特微量分析に関する研究として、意味的類似距離算出と物理的距離による分析手法に関する研究を紹介する。

2.2.1 空間特微量分析による意味的類似距離算出

意味的類似性算出に関する研究開発は機械学習や可視化、探索アルゴリズムなどの分野や情報推薦を取り扱うサービスで広く取り組まれている。Corleyら[1]は、テキストの意味的類似性を測定するための知識ベースの方法を提示しており、従来の字句マッチングに基づくテキスト間の類似距離算出方式よりも単語間の意味的類似に基づくテキスト間の類似距離算出方式が優れていることを示した。本研究では、位置情報の特徴として郵便番号や住所情報を利用し意味的類似距離を算出する点が異なる。Aydoganら[2]は、消費者と生産者との間に発生する要求および供給の交渉で互いのニーズを理解できるようにすることを目的とし、両方が共有オントロジーを使用してサービスを交渉するアーキテクチャを提案し、サービス間の意味的類似距離の算出方式を開発してパフォーマンスを比較している。本研究では、計算コストの削減や意味的・物理的距離の同時算出を

目的としている点が異なる。Júniorら[3]は、k最近傍に基づく協調フィルタリングにおけるアイテム間の意味的距離を計算するための新しい方法を提案しており、映画データセットを用いた評価では、加重リンクのないベースラインと比較した場合、大幅な改善が達成できることを示している。本研究では、郵便番号を用いて地点間の意味的類似距離の算出方式を提案している点が異なる。Piaoら[4]は、LOD対応のレコメンダーシステムで使用できるリソース間のLinked Data意味的距離を計算するためにLinked Data Semantic Distance (LDSD) の基本概念に加えて、さまざまな距離測定を提案しており、リソースとグラフ内のパスのグローバルな外観の両方を使用する正規化を組み込んだ提案手法により、パフォーマンスが大幅に向向上することを示している。本研究では、リソース間ではなく地点間の意味的類似距離の算出を目的としている点が異なる。

2.2.2 物理的距離による空間特微量分析

物理的距離による空間特微量分析に関する研究開発はデータマイニングや探索アルゴリズム、可視化、マッピングなどの分野で広く行われており、Google Map^(注2)に代表されるような地図、地域検索サービスで活用されている。Cuevasら[5]は、地域間のマイニング手法を利用し、Twitterの地理的特性を理解するための包括的な分析を行っており、ユーザの位置情報、地理的関係性、ツイートを含むデータセットを利用し、国ごとに地域のプロファイリングをしている。本研究では、物理的距離の算出に緯度経度ではなく、郵便番号や住所情報を利用している点が異なる。Ahmedら[6]は、移動経路のセットの比較に基づく、2つの平面幾何学的グラフ間の新しい経路ベースの距離測定を提案しており、制限されたリンク長の経路を使用して、グラフ間のグローバルな構造的および空間的な違いが識別できることを示している。本研究では、郵便番号および住所情報から算出される物理的距離の精度の比較として経路ではなく緯度経度(地点)を利用しておらず、計算コストの低減を目的としている点が特異点である。Kieferら[7]は、隠れマルコフモデルに基づく視線マップマッチングアルゴリズムを提案し、視線追跡データセットから地理的特徴ベクトルを生成しマッピングすることを目的としている。本研究では、郵便番号と番地から地理的な特徴をもつベクトルを生成しており、地図生成ではなく物理的距離の新たな算出方式を提案している点が異なる。Kanzaら[8]は、オンラインソーシャルネットワークのユーザー間の地理社会的類似性の新たな距離測度を提案しており、Twitterからのジオタグ付きメッセージを使用した広範な実験では、新しい距離測度が既存の測度よりも正確で効率的であることを示している。本研究では、新たな距離測定方法として郵便番号と番地から物理的精度を算出する点が特異点である。

3. 意味的・物理的距離算出方式

本章では、地点間の意味的・物理的距離の算出方式のプロセスについて述べ、日本と米国の2カ国を対象とする場合の算出方式の相違点について述べる。図2に、従来手法、提案手法

(注2) : <https://www.google.co.jp/maps/?hl=ja>

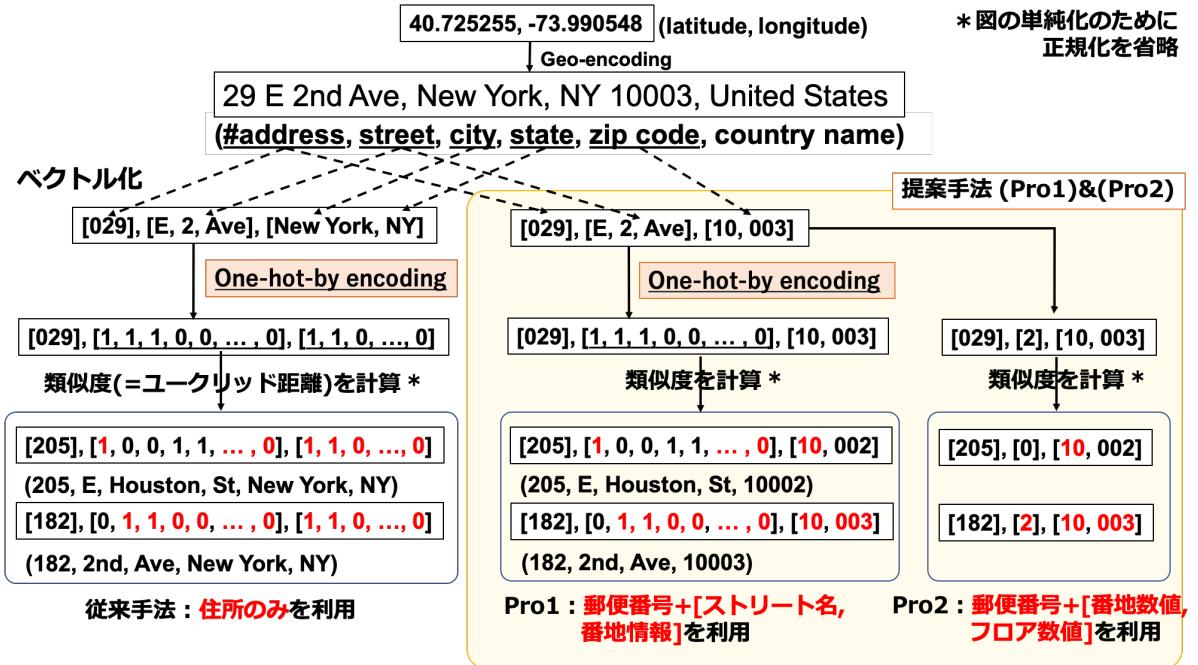


図 2 意味的・物理的距離算出方式(米国)の概要図

* 図の単純化のために正規化を省略

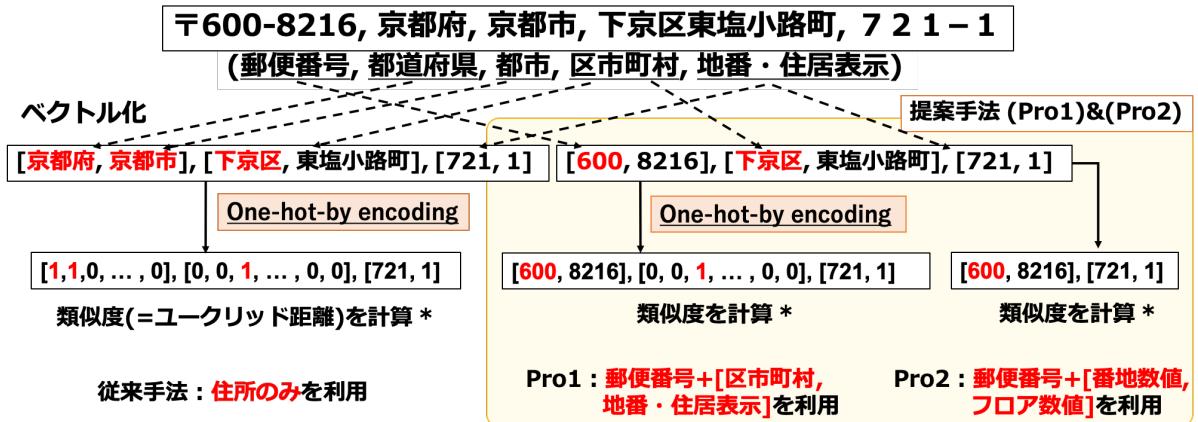


図 3 意味的・物理的距離算出方式(日本)の概要図

(Pro1 および Pro2) の 3 つの手法による意味的・物理的距離算出の流れを示す。従来手法では取得した住所情報から住所のテキストのみを利用するが、提案手法 1(Pro1) では、郵便番号とテキスト(建物名、番地情報、フロア情報)を利用し、提案手法 2(Pro2) では、郵便番号と番地(数値)、フロア情報(数値)を抽出する。

例えば、図 2 の「29 E 2nd Ave, New York」が住所情報として与えられた場合、従来手法では、One-hot encoding により「[029],[1,1,1,0,0,...,0],[1,1,0,...,0]」のように数値以外の情報は 0/1 に変換されるが、州や市、ストリート名などのパターン数に応じて次元数が増加する。これは、Pro1 の手法でも同様だが、こちらは郵便番号(10003)と[番地情報([029])]、ストリート[E,2,Ave]を利用して数値以外の固有単語の出現数を減らし、次元数の増加を抑制している。変換後の特徴ベクトル間の類似度(ユークリッド距離)が POI 間の意味的・物理

的距離として算出される。また、Pro2 の手法では、郵便番号([10,003])と[番地数値([029])]、ストリート番号([2]))のような数値データのみを利用することで、One-hot Encoding を行わずに特徴ベクトル間のユークリッド距離を計算する。

また、日本の場合の意味的・物理的距離算出方式を図 3 に示す。米国の場合と基本的な流れは同じだが、住所の表記方法が異なるため米国のストリートと番地情報に対応する市区町村と地番・住居表示となっており、提案手法 1 では郵便番号(600-8216)と地番・住居表示(721-1)、市区町村(下京区、東塩小路町)]を One-hot 表現したものを利用し、提案手法 2 では郵便番号(600-8216)と地番・住居表示(721-1)から番地数値、フロア数値を抽出する。

また、第 4 章では、米国の都市を対象とした検証を行うため、ここでは日本の 7 枠の郵便番号と米国の 5 枠の郵便番号を利用する場合で異なる点について以下に述べる。

表 1 米国合衆国の郵便番号および POI 数

City	State Code	City Code	#POIs
New York	(NY)	10	001 to 119
Washington DC	(WT)	20	001 to 088
Boston	(BT)	02	101 to 445
San Francisco	(SF)	94	016 to 188
Los Angeles	(LA)	90	001 to 076
Seattle	(ST)	98	101 to 199
			1,215

• 日本の場合

(1) ユーザの行動履歴に含まれる Zipcode を上 3 桁、下 4 桁に分割したものと番地数値、フロア数値などから特徴ベクトルを生成する。例えば、郵便番号が「603-8555」、住所情報として「京都府京都市北区上賀茂本山 14 号館 1 階」が与えられた場合、「[603,8555,0,14,1]」のような特徴ベクトルが生成される。

(2) 生成されたベクトル間のユークリッド距離を算出する。例えば、「[603,8555,0,14,1]」と「[603,8047,339,0,0]」の場合、意味的・物理的距離は 610.8862414558049 となる。

• 米国の場合

(1) ユーザの行動履歴に含まれる Zipcode を上 2 桁、下 3 桁に分割したものと番地数値、フロア数値などから特徴ベクトルを生成する。例えば、郵便番号が「10003」、住所情報として「29 E 2nd Ave, New York, NY 10003, USA」が与えられた場合、「[10,3,29,2]」のような特徴ベクトルが生成される。

(2) 生成されたベクトル間のユークリッド距離を算出する。例えば、「[10,3,29,2]」と「[10,2,205,0]」の場合、意味的・物理的距離は 176.0142039722931 となる。

3.1 郵便番号と住所によるハイブリッド変換方式 (Pro1)

ハイブリッド方式 (提案手法 1 : Pro1) は One-hot encoding により郵便番号と郵便番号以下の住所情報 (テキスト) から特徴ベクトルに変換する。具体的には、図 2 のように「29 E 2nd Ave, NY 10003」から「[29,[1,1,1,0,0,...,0],[1,1,0,...,0]]」のように変換される。住所 (テキスト) のみを利用するのではなく、郵便番号から都道府県/州や市町村区などの POI の大まかな位置情報 (地域性) を表現し、より詳細な位置情報の表現に建物名、番地情報、フロア情報などのテキストを利用する。これにより、従来手法に比べ精度を落とすことなく、計算コストの低減を図る。

3.2 郵便番号と番地等による変換方式 (Pro2)

郵便番号と番地等による変換方式 (提案手法 2 : Pro2) は、郵便番号や番地、フロア数値などの数値データのみから特徴ベクトルに変換する。具体的には、「29 E 2nd Ave, NY 10003」から「[29,2,10,3]」のように変換される。3.1 節で述べたように郵便番号で都道府県および州、市町村区などの POI の地域性を表現するが、Pro1 と異なる点はより詳細な位置情報は番地、フロア数値等の数値データのみで表現する点である。数値データのみを用いるため、従来手法や Pro1 と比べて処理速度やメモリ使用効率の向上が期待できる。また、精度に関しては、第 4 章で従来手法による意味的・物理的距離算出と比較・検証を実施する。

4. 意味的・物理的距離の精度および有効性の検証

本章では意味的・物理的距離の精度および処理速度、メモリの使用効率の評価を行い、提案手法の有用性の検証を行う。評価のために米国の 6 都市 (ニューヨーク、ワシントン、ボストン、サンフランシスコ、ロサンゼルス、シアトル) から 6,720 件の POI データを取得した。取得した米国の各都市データを、表 1 に示す。

4.1 距離精度の評価方法

まず、物理的距離の精度を評価するために各 POI 間の実距離を緯度経度からユークリッド距離を算出し、Min-Max 正規化を行い、実距離と従来手法、実距離と提案手法との平均二乗誤差 (MSE) より評価する。また、意味的距離の評価は、各 POI 間のコサイン類似度を算出し、Min-Max 正規化を行い、従来手法と提案手法との MSE より評価する。特に、実距離、意味的・物理的距離の算出は、各検証により以下の 3 つの方法で行われる。それぞれの概要図を図 4 に示す。

(1) 1 つの POI (ユーザが選択した地点、または現在位置) とそれ以外の POI との実距離、意味的・物理的距離を算出。都市に含まれる POI の数を N とすると、N - 1 個の POI 間の距離を計算する。

(2) 全ての POI 間の実距離、意味的・物理的距離を総当たりで算出。都市に含まれる POI の数を N とすると、 $N(N-1)/2$ 個の POI 間の距離を計算する。

(3) 都市間の POI の実距離、意味的・物理的距離を総当たりで算出。都市 A に含まれる POI 数を X、都市 B に含まれる POI 数を Y とすると、 $X \times Y$ 個の POI 間の距離を計算する。

また、本実験では、従来手法である「住所のみを用いる場合 (a)」と提案手法である「郵便番号とストリート名、番地情報を用いる場合 (b)」、そして「郵便番号と番地数値を用いる場合 (c)」の 3 つの手法を検証する。

4.1.1 物理的距離の比較

表 2 は (1) の方法 (図 4) で実距離と意味的・物理的距離の類似度を比較した結果である。各手法の MSE の平均をみると (c) の手法が 0.1131 で最も低い値を示しており、この表 2 からは (c) の手法が最も精度が高いことがわかる。しかしながら、(c) の各都市の MSE をみると、NY は 0.0137 で全ての結果の中で最も低い値であるが、SF は 0.2711 で高い値を示していることがわかる。また、表 3 は (2) の方法 (図 4) で実距離と意味的・物理的距離の類似度を比較した結果であるが、こちらも表 2 と同様に (c) の手法が最も高い精度を示しており、NY の 0.0132 が最も低い値で、SF の 0.1665 が高い値を示している。これらの結果で特筆すべきことは、提案手法による都市内の POI 間の物理的距離算出は (c) の手法が最も精度が良いということである。また、(c) の MSE の値で NY が最小値、SF が最大値となった原因としては、NY 市内では郵便番号間の数値的な近さと実距離の類似性が高い関係にある POI が多いのに対し、SF 市内は郵便番号間の数値的な近さと実距離の類似性が低い関係にある POI が多いため、誤差が大きくなつたからだと考えられる。



図 4 実距離、意味的・物理的距離の算出方法

4.1.2 意味的距離の比較

表 4 は(1)の方法(図 4)で各手法で意味的距離を算出し、従来手法(a)と提案手法(b), (c)のMSEを算出した結果である。全体的な結果として各手法のAverageをみると、(b)が 0.0347 、(c)が 0.0516 で(b)の方が低い値を示しており、従来手法(a)との類似性が高いことを確認できた。また、表 5 は(2)の方法(図 4)で各手法で意味的距離を算出し、(a)と(b), (c)のMSEを算出した結果であるが、この結果でも同様に(b)が 0.0559 、(c)が 0.0728 で(b)の方が低い値を示しており、(a)と(b)の類似性が高いことが確認できた。これらの結果は、(b)の手法による意味的距離算出の精度が(c)よりも良いことを意味しており、これは(a)と(b)の共通処理であるOne-Hot Encodingが地点間の意味的距離の精度に大きく貢献しているからだと考えられる。ただ、(c)の手法も(b)とのMSEの差はわずか 0.0169 であるため、高い精度を示していると言える。

4.1.3 都市間の物理的距離の比較

4.1.2節では、提案手法で算出した意味的距離が高い精度であることを確認できたが、4.1.1節では、物理的距離は都市によってMSEの値にばらつきがあった。ここでは2つの都市間(例えば、NYとWT)を対象とすることで、より離れたPOI間の物理的距離算出の精度を検証する。表6、表7、表8は(a), (b), (c)の各手法で(3)の方法(図4)により都市間の意味的・物理的距離を算出し、実距離と比較した結果である。表6では、(a)のMSEの全体平均は 0.0500 と小さな値を示し、物理的距離の精度が高いことを確認できた。一方で、表7と表8のMSEの全体平均は 0.1370 と 0.1875 で大きな値を示し、従来手法(a)よりも提案手法(b), (c)の精度が低い結果となった。(b)の手法は(a)と同様にOne-Hot Encodingを利用しているにも関わらず、(a)と(b)のMSEの全体平均の差は 0.087 であった。これは、郵便番号から生成される2次元のベクトルがPOIの詳細な位置を十分に表現できていないことが原因と考えられる。この結果から、郵便番号の上2桁が異なるような離れた地点間の物理的距離算出に現時点では提案手法は適しておらず、実距離と物理的距離のMSEが大きい場合は、誤差が小さくなるように郵便番号を補正する必要がある。

4.2 処理速度の評価

意味的・物理的距離の有効性を検証するために(a),(b),(c)のそれぞれについて処理速度の比較を行う。具体的には、6都市のそれぞれについて100件ずつデータ数(=POI数)を増やし

表 2 緯度経度の実距離と従来手法(a)および提案手法(b), (c)の類似度の比較

City	Method		
	(a)	(b)	(c)
(NY)	0.2378	0.0141	0.0137
(WT)	0.1076	0.1612	0.0831
(BT)	0.0497	0.0587	0.1256
(SF)	0.0688	0.0626	0.2711
(LA)	0.0350	0.2144	0.0595
(ST)	0.3814	0.3110	0.1256
Average	0.1467	0.1370	0.1131

表 3 緯度経度の実距離と従来手法(a)および提案手法(b), (c)の類似度の総当たりによる比較

City	Method		
	(a)	(b)	(c)
(NY)	0.2027	0.0154	0.0132
(WT)	0.0803	0.1057	0.0418
(BT)	0.0709	0.0805	0.0548
(SF)	0.0675	0.0672	0.1665
(LA)	0.1078	0.1072	0.0384
(ST)	0.1697	0.0489	0.0917
Average	0.1165	0.0708	0.0677

表 4 従来手法(a)と提案手法(b), (c)の類似度の比較

City	Method	
	(b)	(c)
(NY)	0.0455	0.0441
(WT)	0.0329	0.1031
(BT)	0.0443	0.0241
(SF)	0.0285	0.0599
(LA)	0.0286	0.0421
(ST)	0.0285	0.0365
Average	0.0347	0.0516

ながら(1)の方法で意味的距離を算出した場合の各手法の処理速度を測定した。本実験では、pythonのtimeライブラリを用いてMin-Max正規化による意味的・物理的距離の算出に要する時間を測定する。

6都市の意味的・物理的距離算出の処理速度の測定結果を図5に示す。図5をみると、全体的な結果として(a), (b)は全都市でほぼ同じ処理速度を示し、(c)が最も速い処理速度を示した。また、グラフの傾向から、データ数の増加に伴い(a), (b)

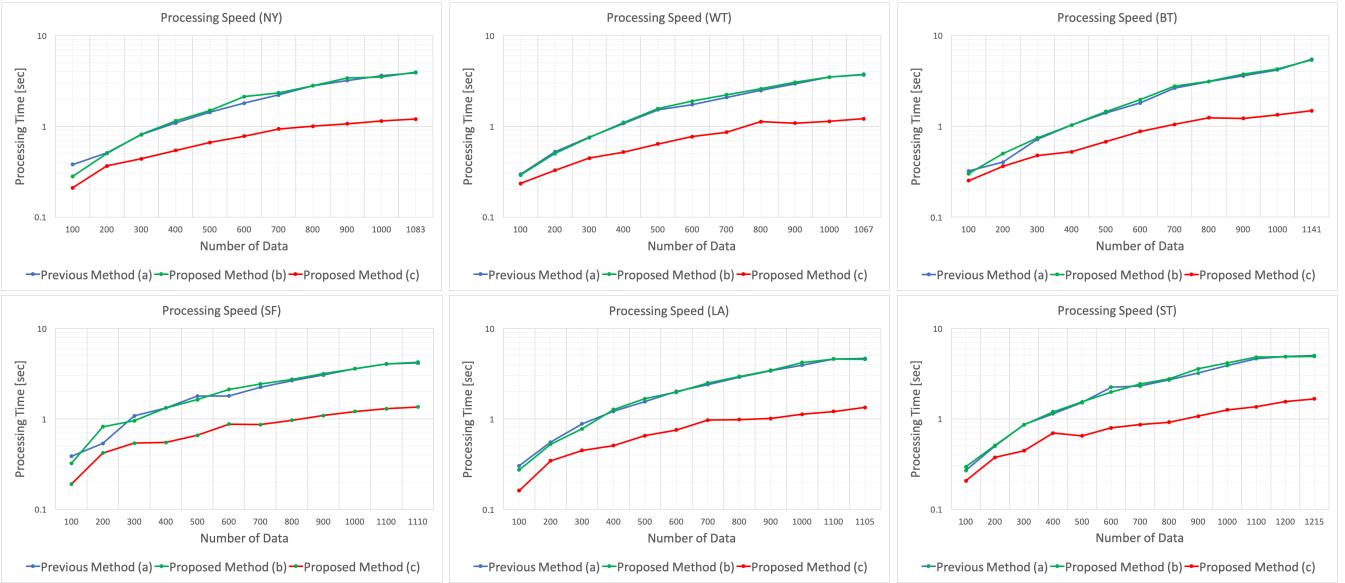


図 5 従来手法 (a) および提案手法 (b, c) による各都市ごとの意味的・物理的距離算出の処理速度の比較

表 5 総当たりによる従来手法 (a) と提案手法 (b), (c) の類似度の比較

City	Method		MSE		Average
	(b)	(c)			
(NY)	0.0503	0.0853			
(WT)	0.0590	0.0909			
(BT)	0.0600	0.0671			
(SF)	0.0397	0.0660			
(LA)	0.0643	0.0694			
(ST)	0.0602	0.0582			
Average	0.0559	0.0728			

表 6 従来手法 (a) による都市間の類似度算出結果

City	Method						Average
	(NY)	(WT)	(BT)	(SF)	(LA)	(ST)	
(NY)		0.0814	0.0476	0.0618	0.0830	0.0905	0.0729
(WT)	0.0814		0.0308	0.0408	0.0458	0.0530	0.0504
(BT)	0.0476	0.0308		0.0343	0.0313	0.0291	0.0346
(SF)	0.0618	0.0408	0.0343		0.0343	0.0426	0.0428
(LA)	0.0830	0.0458	0.0313	0.0343		0.0431	0.0475
(ST)	0.0905	0.0530	0.0291	0.0426	0.0431		0.0517
Average	0.0729	0.0504	0.0346	0.0428	0.0475	0.0517	0.0500

と (c) の処理速度の差がより大きくなることを確認でき、最も差が拡がった BT の 1141 件では 72.4%ほど低減されている。これは、(a) と (b) の手法は意味的・物理的距離算出の前処理として住所やストリート名などのテキストデータの One-hot Encoding が必要となるが、(c) はそのような前処理を必要とせず、郵便番号と番地数値などの数値データから直接意味的・物理的距離を算出するため (a), (b) に比べ高速に計算可能であるからだと考えられる。これらの結果から、(c) の処理速度が最も速く、膨大なデータに対しても (c) が最適であることが示された。

表 7 提案手法 (b) による都市間の類似度算出結果

City	Method						Average
	(NY)	(WT)	(BT)	(SF)	(LA)	(ST)	
(NY)		0.0909	0.2147	0.0451	0.0878	0.0410	0.0959
(WT)	0.0909		0.1414	0.1702	0.1947	0.1119	0.1418
(BT)	0.2147	0.1414		0.2867	0.1515	0.2271	0.2043
(SF)	0.0451	0.1702	0.2867		0.0570	0.1871	0.1492
(LA)	0.0878	0.1947	0.1515	0.0570		0.0485	0.1079
(ST)	0.0410	0.1119	0.2271	0.1871	0.0485		0.1231
Average	0.0959	0.1418	0.2043	0.1492	0.1079	0.1231	0.1370

表 8 提案手法 (c) による都市間の類似度算出結果

City	Method						Average
	(NY)	(WT)	(BT)	(SF)	(LA)	(ST)	
(NY)		0.0479	0.3910	0.1254	0.0520	0.0937	0.1420
(WT)	0.0479		0.1190	0.2652	0.0962	0.1832	0.1423
(BT)	0.3910	0.1190		0.2899	0.1220	0.2136	0.2271
(SF)	0.1254	0.2652	0.2899		0.3341	0.2781	0.2585
(LA)	0.0520	0.0962	0.1220	0.3341		0.2006	0.1610
(ST)	0.0937	0.1832	0.2136	0.2781	0.2006		0.1938
Average	0.1420	0.1423	0.2271	0.2585	0.1610	0.1938	0.1875

4.3 メモリ使用効率の評価

意味的距離の有効性を検証するために (a), (b), (c) のそれぞれについてメモリ使用効率の比較を行う。本実験では、memory profiler と呼ばれる python モジュールを用いてプログラムのメモリ消費量の行単位の分析を行い、各手法の意味的距離算出のメモリ使用効率を測定する。

図 6 に (1) の方法で POI(6720 件) を含む全都市データに対して意味的距離を算出した場合の各手法のメモリ使用効率の比較結果を示す。図 6 をみると、全体的な結果として (a), (b) は約 2300MiB のメモリを約 230 秒間使用し続けたのに対し、(c) は同じメモリ使用量で約 1/10 の 25 秒ほどで処理を完了させていることがわかる。この結果から、(c) は (a), (b) よりもメモ

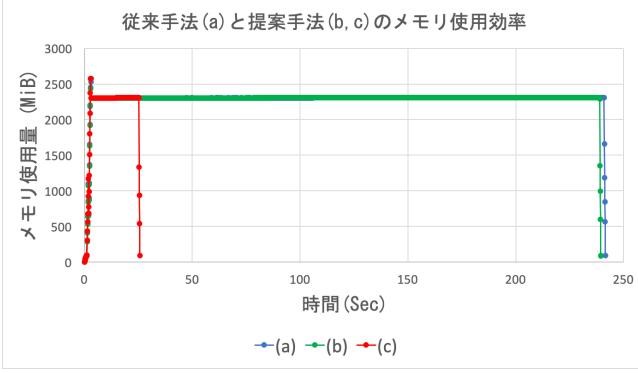


図 6 意味的・物理的距離算出のメモリ使用効率の比較

り使用効率が 90%ほど高く、最も良い結果を示したことを確認できた。また、(a), (b) のメモリ使用効率が悪い原因としては、4.2 章で確認した One-hot Encoding によるテキストの特徴ベクトル間の距離算出に時間がかかることが考えられ、(c) のメモリ使用効率が良い原因としては、4.2 章で確認した意味的・物理的距離算出に数値データのみを利用するため処理が高速であるからだと考えられる。メモリ使用効率は処理速度に依存するため、扱うデータが膨大であればある程 (a), (b) と (c) の差は顕著になると考えられる。

4.4 word2vec による意味的・物理的距離算出精度および処理速度の比較

文章のベクトル化における次元数の削減を考慮した代表的な自然言語処理に word2vec や fasttext, BERT 等がある。本論文では、住所情報のベクトル化における従来手法として One-Hot Encoding を挙げていたが、本節では、word2vec を従来手法とした場合の意味的・物理的距離の精度および処理速度を検証し、提案手法 (b), (c) と比較する。

表 9 は (2) の方法で (b), (c) および word2vec で算出した物理的距離と実距離を MSE で比較した結果で表 10 は (2) の方法で (b), (c) および word2vec で算出した意味的距離を MSE で比較した結果である。表 9 は全体的に word2vec の MSE の値が小さく、word2vec の平均値は 0.0450 で全手法の中で最も高い精度を示したが、(c) との差はわずか 0.022 であった。また、表 10 では、(b) と (c) の平均値を比較すると、(c) が 0.0763 で (b) よりも小さい値を示し、(b) よりも (c) の精度が高い結果となつた。以上の結果から、提案手法 (b, c) は word2vec と比較すると、物理的距離の精度に大きな差はないが、都市ごとに値にバラツキがあるため分散を抑える必要がある。また、意味的距離の精度は (c) が高いが、ST で誤差が大きいため、実用上は物理的距離と同様に分散を抑える必要がある。

図 7 は (b), (c) および word2vec の処理速度を比較した結果である。各手法の処理速度は (c) が最速で、2番目が word2vec, 3番目が (b) となった。(c) で生成されるベクトル (3 次元) に対し、word2vec の単語の分散表現は 200 次元、(b) の One-Hot ベクトルは数千次元にも及ぶため、次元数が多いほど計算量も多くなることから図 7 の結果になったと考えられる。

表 9 word2vec, (b), (c) の物理的距離の比較

City \ Method	MSE		
	(b)	(c)	word2vec
(NY)	0.0154	0.0132	0.0457
(WT)	0.1057	0.0418	0.0416
(BT)	0.0805	0.0548	0.0359
(SF)	0.0672	0.1665	0.0647
(LA)	0.1072	0.0384	0.0384
(ST)	0.0489	0.0917	0.0435
Average	0.0708	0.0677	0.0450

表 10 word2vec と (b), (c) の意味的距離の比較

City \ Method	MSE	
	(b)	(c)
(NY)	0.0584	0.0578
(WT)	0.1091	0.0480
(BT)	0.1125	0.0710
(SF)	0.0534	0.0978
(LA)	0.0895	0.0534
(ST)	0.0683	0.1299
Average	0.0819	0.0763

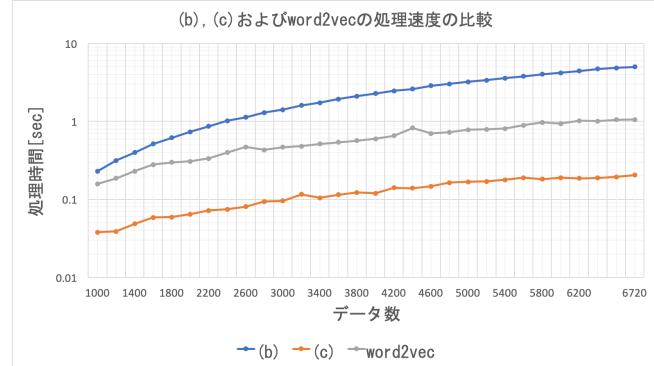


図 7 Pro2 と word2vec の処理速度の比較

5. まとめ

本論文では、提案手法 (Pro1) の「郵便番号と住所によるハイブリッド変換方式」および提案手法 (Pro2) の「郵便番号と番地等による変換方式」の 2 つの手法に基づく意味的・物理的距離の算出方法を提案し、米国の都市データを利用し、従来手法、Pro1, Pro2 の各手法で精度や処理速度、メモリ使用効率を比較することで有効性の検証を行った。

精度の検証では、都市内の POI 間の意味的・物理的距離は Pro1, Pro2 が特に高い精度を示したが、都市間の意味的・物理的距離は従来手法が高い精度を示した。これらの結果から、都市内 (郵便番号の上桁が同じ場合) という条件下では Pro1, Pro2 の有用性が高いことを確認できた。また、処理速度とメモリ使用効率を測定した結果、Pro2 は One-Hot Encoding が不要であり、数値データのみを扱うことから処理速度が他の手法に比べ約 3 倍速く、メモリ使用効率も処理が高速なため 90%ほど高いことがわかった。また、One-Hot Encoding よりも次元数を削減可能である word2vec との比較結果では、word2vec は物理的距離と意味的距離の精度は共に良かったが Pro1, Pro2

との差は大きくなく、処理速度は Pro2 が最速となった。これらの検証結果から、提案手法 (Pro1, Pro2) は都市内のような物理的な距離の近い地点間では意味的距離・物理的距離の精度は常に高く、また、Pro2 は計算コストを最も抑制可能であるため、膨大なデータに対する Pro2 の有用性が高いことを示した。

今後は、広範囲の POI 間の意味的・物理的距離について提案手法の精度向上を目指し、地点間の郵便番号を用いた適切なベクトル表現の決定法について検討する。また、日本の郵便番号でも検証を実施する予定である。さらに、意味的・物理的距離を利用した機械学習による地域ごとのユーザ特性抽出への応用を検討している。

謝 辞

本研究の一部は、株式会社 J&J 事業創造および JSPS 科研費 JP19H04118, 19K12240, ならびに京都産業大学先端科学技術研究所（HMD 共生科学研究センター）の助成を受けたものである。ここに記して謝意を表す。

文 献

- [1] Courtney Corley and Rada Mihalcea. Measuring the semantic similarity of texts. In *Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, EMSEE '05, p. 13–18, USA, 2005. Association for Computational Linguistics.
- [2] Reyhan Aydogan and Pinar Yolum. Learning consumer

preferences using semantic similarity. In *Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS '07, New York, NY, USA, 2007. Association for Computing Machinery.

- [3] Salmo M.S. Júnior and Marcelo G. Manzato. Collaborative filtering based on semantic distance among items. In *Proceedings of the 21st Brazilian Symposium on Multimedia and the Web*, WebMedia '15, p. 53–56, New York, NY, USA, 2015. Association for Computing Machinery.
- [4] Guangyuan Piao and John G. Breslin. Measuring semantic distance for linked open data-enabled recommender systems. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, SAC '16, p. 315–320, New York, NY, USA, 2016. Association for Computing Machinery.
- [5] Ruben Cuevas, Roberto Gonzalez, Angel Cuevas, and Carmen Guerrero. Understanding the locality effect in twitter: Measurement and analysis. *Personal Ubiquitous Comput.*, Vol. 18, No. 2, p. 397–411, February 2014.
- [6] Mahmuda Ahmed, Brittany Terese Fasy, Kyle S. Hickmann, and Carola Wenk. A path-based distance for street map comparison. *ACM Trans. Spatial Algorithms Syst.*, Vol. 1, No. 1, July 2015.
- [7] Peter Kiefer and Ioannis Giannopoulos. Gaze map matching: Mapping eye tracking data to geographic vector features. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '12, p. 359–368, New York, NY, USA, 2012. Association for Computing Machinery.
- [8] Yaron Kanza, Elad Kravi, Eliyahu Safra, and Yehoshua Sagiv. Location-based distance measures for geosocial similarity. *ACM Trans. Web*, Vol. 11, No. 3, July 2017.