未登録 PoI 探索のための 時系列ソーシャルメディアデータに基づく位置推定手法の提案と評価

澤野耕平 [†1]
奈良先端科学技術大学院大学

松田裕貴 [†2]
岡山大学

中谷響 [†3]
奈良先端科学技術大学院大学

大内啓樹 [†4]
奈良先端科学技術大学院大学

諏訪博彦 [†5]
奈良先端科学技術大学院大学

安本慶一 [†6]
奈良先端科学技術大学院大学

1. はじめに

近年，ソーシャルメディアの普及により観光地にて観光客がソーシャルメディア投稿を行う機会は多い．観光客はソーシャルメディアを用いて旅行体験を投稿している．このような使用が行われる中で，ソーシャルメディアには観光客が興味を引く場所である Point of Interest（PoI）に関する情報が多く集積している．この集積した情報があるソーシャルメディアを一種の観光情報データベースとし，観光地での旅程選定の参考にするような情報検索ツールとしての使用もされている [1]．具体的には Instagram の「地図」機能である．この機能は現在位置周辺の位置情報付与された投稿を表示することができ，これを用いて観光客が近隣の PoI を検索することができる．このようにソーシャルメディア投稿が言及している位置を推定することは観光地での意思決定において重要である．

ソーシャルメディアの投稿データに位置情報を付与する方法には，デバイスの GPS データを利用するものと，投稿に対して地図データベース上の PoI をタグ付けるものがあるが，これらの方法にはそれぞれ課題が存在する．前者は，GPS 制度に大きく依存するという点や，実際に GPS データが付与されている投稿が全体の 0.4% ほどとごく少数という点が挙げられる．後者は，既存の地図データベース（地理 DB）を参照するためにデータベースに載っていない PoI には対応できないという点が挙げられる．Instagram の「地図」機能は地図データベース上に存在する PoI をタグ付けする機能を使用して既存の地理 DB（Google Map や OSM など）に登録されている位置情報を参照して投稿に位置情報を埋め込んだ物を地図上に可視化した機能である．

既存の PoI 位置推定手法は投稿した地点の画像や投稿データに付与された位置情報を入力データとして使用している．また，既に地理 DB に登録されている PoI をクラスとし，投稿をクラス分類することで位置推定を行っている．位置情報の有無や地理 DB の登録された範囲に位置推定が限定されており，位置情報を付与している投稿データが少ない中で，これらの既存手法を適用するのは制限されたデータに対してしか行うことができない．

そこで本研究では観光地でのソーシャルメディア投稿から観光客の時系列での移動経路と相対的な位置関係を用いることで，既存の地図データベースの範囲や投稿の位置情報に依存することなく，観光客にとって関心の高い場所である PoI の位置を推定する手法を提案する．

2. 関連研究

2.1. 文章から位置推定する手法（ジオパージング）

文章中に含まれる場所に対する位置推定の既存の手法は大きく二つに分けられる．一つは文章中のテキストから地名を抽出し GeoNames や OSM などの地理 DB にある場所にクラス分けをする分類問題としてのアプローチである．もう一つは文章中のテキストから緯度経度を回帰する回帰問題としてのアプローチである．分類問題としての一般的なアプローチとしてこれまでには自然言語による研究がされてきた [2, 3, 4]．これらの研究では地球表面をグリッド分割し，テキストの地名が属するグリッドを予測することで，位置推定を行っている．

ソーシャルメディア投稿のテキストを用いた研究として，Li らは，ユーザの自由記述テキストと投稿プラットフォームといったカテゴリカルデータを BERT ベースのエンコーダで埋め込み，時間情報を階層型に分割しそれぞれエンコーディングすることで，これらのベクトル表現の特徴間の相関を学習させ，最終的に PoI が地理データベースの中の地理

Location Estimation Method using Time-Series Social Networking Service Data for Exploring Unregistered PoI

[†1] KOHEI SAWANO, Nara Institute of Science and Technology
[†2] YUKI MATSUDA, Okayama University
[†3] HIBIKI NAKATANI, Nara Institute of Science and Technology
[†4] HIROKI OUCHI, Nara Institute of Science and Technology
[†5] HIROHIKO SUWA, Nara Institute of Science and Technology
[†6] KEIICHI YASUMOTO, Nara Institute of Science and Technology

エンティティに属するかの確率を出力している．時間情報の階層的な表現により投稿の時間的なパターンを捉えやすくなり，既存の手法に比べて推定精度が向上したと報告している [5]．これらの分類問題としてのアプローチは，既存の GeoNames や Wikipedia といった地名データベースの範囲でしか分類ラベルを作れないという制限が存在している．

直接テキストから緯度経度を推定する回帰問題としてのアプローチの一つは，確率モデルを出力することによるアプローチである [6, 7]．Benjamin らは wikipedia データで学習し，出力に混合分布による空間的な確率分布を出力する．出力される確率分布は予測される緯度経度を中心としており，複数の分布が予測結果として出力された場合には混合分布が付与されることで，曖昧な地理表現などにも柔軟に対応可能なことを示している [6]．

2.2. ソーシャルメディア投稿データを用いた位置推定

Dai らは，ソーシャルメディア投稿のテキストとハッシュタグなど複数のデータを用いるマルチモーダル表現学習フレームワーク（MRLF）を提案している．このフレームワークは画像，テキスト，ハッシュタグの間の特徴を相互に学習することで，位置推定の向上がされたと報告している．特に投稿の PoI に属する分類精度では平均 8% の精度向上を実現し，分類された PoI と実際の投稿位置との間の距離では平均誤差約 900m を実現している [8]．

2.3. 本研究の位置付け

既存研究の分類手法は GeoNames などの地理 DB のエンティティをラベルとして使用するものであるため，地理 DB にないエンティティは位置推定できない．また，回帰手法については学習にジオタグ付きのソーシャルメディアデータを使用しており，このようなデータを得ることは難しいということ，既存のソーシャルメディアデータの持つジオタグは地理 DB に依存したものなので，回帰手法においても地理 DB の範囲による制限は存在している．また，既存のテキストから直接，緯度経度を回帰している手法の誤差は数十 km オーダーであり，PoI レベルの位置推定への応用は難しいという課題がある．これらの問題点を踏まえて，本研究では観光地での時系列ソーシャルメディア投稿を用いた PoI 位置推定手法を提案する．観光地においてソーシャルメディアユーザは興味を引いた地点についての投稿を行いながら観光していると仮定する．この時系列の投稿列には既存のジオコーディングモデルでは位置推定できないような PoI についての投稿も存在している（例．地理 DB に存在しないかもしくは地理 DB に存在するあるエリア内のポイントなど）．ジオコーディング可能な PoI を Geocoding PoI（GPoI），不可能な PoI を Target PoI（TPoI）と定義し，TPoI と GPoI の時系列的関係から，移動可能範囲を算出しそれらを複数組み合わせることで PoI レベルのジオコーディングの実現を目指す．

3. 位置推定フレームワーク

本章では，まずはじめに我々の提案する時系列ソーシャルメディア投稿を用いた位置推定フレームワークの概要について説明する．その後，本研究の主な貢献となる位置推定アルゴリズムの概要と詳細な構成について説明する．

3.1. 提案フレームワークの概要

本節では，本研究が提案する位置推定フレームワークの概要と構成について述べる．本研究は観光地において観光客が目についたものをリアルタイムでソーシャルメディアに投稿しながら観光しているという過程を置いた上で，いくつかの投稿が言及する PoI をアンカーとしてアンカー投稿と非アンカー投稿（位置推定の対象となる投稿）の時系列関係から移動可能範囲を推定しその範囲をユーザごとに重ね合わせ，最も重なった部分の重心を推定ポイントとして位置推定することを目的とする．

3.2. データ収集

本節では，提案手法で用いたデータ収集のためのソーシャルメディアプラットフォームについて述べる．

提案フレームワークでは，データ収集実験に用いるソーシャルメディアとして松田らのレポっと [9] を用いる．レポっとは，スマートフォンアプリケーションであり，写真を撮影し，その写真に対してコメントを投稿するアプリケーションである．このレポっとを用いて実験参加者に対して観光地において観光しながら目に留まったものを写真撮影し，投稿してもらうという実験を行った．

3.3. ジオコーディング

本節では，提案フレームワークの中のテキストから緯度経度を推定するモデルについて説明する．

ジオコーディングに使用するモデルは，中谷ら [10] が提案している BERT ベースのジオコーディングモデルを用いる．中谷らは，自由記述文章を入力としたジオコーディングタスクが抱える課題である地名の曖昧性解消のために，文章中に登場する地名表現部分だけでなく前後の文章を含めたベクトル化と地理 DB である OSM のエントリをそれぞれ異なるエンコーダでベクトル化することを提案している．このモデルは，入力として受け取ったテキストに対して緯度経度のペアを出力するが，出力結果は推定確率の高い順にランク形式になっている．本研究では，最も確率の高い

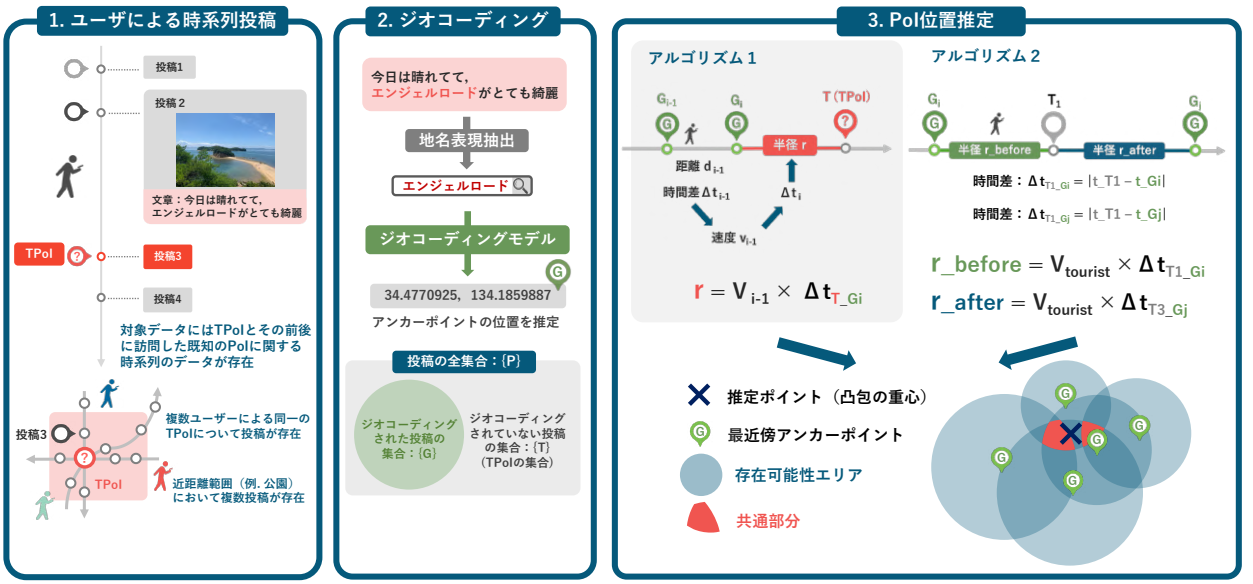

図 1　提案フレームワークの全体図

上位 1 位の推定緯度経度を用いる．

3.4.　位置推定アルゴリズム

本節では，提案フレームワークのうち，アンカーとなる投稿を用いた位置推定アルゴリズムについて説明する．本研究では，位置推定について二つのアルゴリズムを実装し，その精度を評価した．手法 1 は，位置推定対象の PoI（Target PoI：TPoI）に関する投稿とその直前および直後のジオコーディングにより緯度経度が推定された 2 投稿から，TPoI 訪問前後のユーザの局所的な移動速度を算出しこの移動速度を用いてユーザの移動可能範囲を算出するものである．手法 2 は全ユーザの移動軌跡から算出した観光地での平均移動速度を各ユーザの移動速度として近似的に用いて移動可能範囲の算出を行うものである．手法 1,2 において算出したユーザの移動可能範囲内の TPoI の存在確率は一様分布に従うと仮定している．次節からそれぞれの具体的なアルゴリズムについて詳細を説明する．

3.4.1.　変数定義

本節では提案する手法 1,2 で用いるデータ集合の定義を行う．本研究の実験で得られたソーシャルメディアの投稿の集合とその要素を，

$$P = \{\, p_j \mid \forall j \in Z \},\quad p_j = \begin{pmatrix} \text{caption}, \\ \text{timestamp}, \\ \text{latitude}, \\ \text{longitude}, \\ \text{geo latitude}, \\ \text{geo longitude}, \\ \text{cos similarity} \end{pmatrix}, \tag{1}$$

- latitude, longitude：レポっとで投稿した時に自動で付与される緯度経度（正解データとしても用いる）
- geocoding latitude, geocoding longitude, cosine similarity：ジオコーディングモデルによって推定された緯度経度とコサイン類似度（ジオコーディングされた投稿のみ値を持ち，それ以外の投稿は NaN）

と定義する．また，集合 G をジオコーディングモデルによってジオコーディングされた投稿の集合と定義する．このとき，

$$G \subseteq P, \tag{2}$$

である．また，集合 P の要素かつ，集合 G の要素でない全ての投稿の集合を T と定義する．

$$T = \{p \in P \mid p \notin G\},\quad G \cap T = \emptyset \tag{3}$$

3.4.2.　手法 1：時系列ウィンドウを用いた局所速度による位置推定

手法 1 では推定対象の直前および，直後の 2 投稿 ($p \in G$) と TPoI に関する投稿 ($p \in T$) の 5 投稿を 1 単位とする時系列のウィンドウを定義し，ウィンドウ内で局所速度を計算

し位置推定を行う．この手法の詳細についてアルゴリズム全体の流れを疑似アルゴリズムとして示し（Algorythm1），その中で使用する各関数の詳細について以下に示す．

Algorithm 1 位置推定全体フロー

```
df ← ReadRawData(DataPath)
局所速度計算を行い，PoI クラスごとの円を算出
PoiClassDict = {}
for i ∈ length(df) do
    if i mod 5 = 0 then
        window_df = MakeWindow(df)
        p_{-2} = window_df[0] ∈ G
        p_{-1} = window_df[1] ∈ G
        p_0 = window_df[2] ∈ T
        p_{+1} = window_df[3] ∈ G
        p_{+2} = window_df[4] ∈ G
        v_± = ComputeVelocity(p_{±2}, p_{±1})
        center_list = p_{±1}
        Δt_± = TimeDiff(p_{±1}, p_0)
        radius_list = ComputeRadius(v_±, Δt_±)
        center_list = p_{±1}
        PoiClassDict[p_0["PoiClass"]] = (center_list, radius_list)
    end if
    window_df.delete(0)        ▷ ウィンドウをスライド
end for
グリッドマップを計算し，最も高い値を持つエリアを抽出
GridMap_dict = BuildGridmap(PoiClassDict)
for gridmap ∈ Gridmap_dict do
抽出したエリアの凸包を計算したのちに，重心を計算
    Centeroid = ComputeCenteroid(gridmap)
計算した重心の値と正解位置のユークリッド距離を計算
    Result = EuclideanDistance(Centeroid, GroundTruth)
    ResultTable.add(Result)
end for
return ResultTable
```

1. **時系列ウィンドウの構築と処理:Algorythm1，6 行目**
   - **MakaWindow** 関数で投稿のインデックスを順次 *window_list* に追加する．
   - *window_list* の長さが 5 になった時点で，中央のインデックスを参照し，前後 2 行を含む計 5 行のデータ *df* を取得し，window_df として返す．
2. **速度および半径の計算:Algorythm1，7-20 行目**
   - window_df の各投稿を $p_{-2}, p_{-1}, p_0, p_{+1}, p_{+2}$ とする．$(p_{-2}, p_{-1}, p_{+1}, p_{+2}) \in G$, $p_0 \in T$
   - 各投稿間の時間差を $\Delta t_{\pm 1}$ と定義する．

$$\Delta t_{\pm 1} = \textbf{TimeDiff}(p_{\pm}1, p_0) \tag{4}$$

   - 各投稿の位置情報から距離を計算し，**局所速度** $v_{\pm}$，半径 $r_{\pm}$ を算出する．（※ **ComputeVelocity** は 2 投稿から局所速度を計算する関数）

$$v_{\pm} = \textbf{ComputeVelocity}(p_{\pm 2}, p_{\pm 1}) \tag{5}$$

$$r_{\pm} = v_{\pm} \times \Delta t_{\pm 1} \tag{6}$$

3. **円の計算とグリッドマップの生成:Algorythm1，21-22 行目**
   $p_{\pm}$ を中心とし半径 $r_{\pm}$ の範囲をユーザ移動可能範囲とする．各移動可能範囲に対して，範囲内の各ピクセルの値を以下で定義する任意の重み (w) に更新する (**BuildGridmap**).
   ここで取りうる重み (w) を以下のように定義する．

$$w = \begin{cases} 1, & \text{count の場合,} \\ \pi r^2 & \text{area の場合} \end{cases} \tag{7}$$

$$H(x, y) = H(x, y) + w \tag{8}$$

   重みは 2 つの重みを定義した．一つは単純に移動可能範囲が最も重なったところに TPoI が存在する可能性が高いという考えのもとで，移動可能範囲の重なった回数による重み（count）を定義した．もう一つは移動可能範囲の大きさも考慮に入れた重みで，移動可能範囲が大きい場合には存在する範囲が大きくなり，推定の確信度が低くなるという考えから，移動可能範囲の面積の逆数を重みとする重み（area）を定義した．
4. **推定ポイントの抽出と結果の出力:Algorythm1，24-29 行目**
   - グリッドマップの各セルから最も値の高いセルを特定し，そのセルを全て内包するような凸包を算出する．(**ComputeCenteroid**)
   - 凸包の重心を算出し，重心と正解位置とのユークリッド距離を計算する．(**EuclideanDistance**)

3.4.3. 手法 2：平均的観光客移動速度による位置推定

アルゴリズム 1 が時系列ウィンドウにより局所的なユーザの移動速度を個別に計算し，位置推定を行っていたのに対して，アルゴリズム 2 では，観光中の観光客の歩行速度は平均速度で近似できるという仮定のもと，ユーザ全体の移動軌跡から観光地での平均歩行速度を求め，それを用いた移動可能範囲を計算し位置推定を行うものである．観光客の平均的な移動速度を用いた移動可能範囲計算による位置推定フローの概要を以下に示す．

Algorithm 2 位置推定全体フロー

```
df = ReadRawData(DataPath)
ステップ 1: 全ユーザの平均移動速度を計算
v_list = []
for user_df ∈ df.groupby(user) do
    for i ∈ user_df do
        v_i = ComputeVelocity(user_df[i], user_df[i+1])
        v_list.add(v_i)
    end for
end for
V_tourist = v_list.average
ステップ 2:POI クラスごとの円情報を計算
PoiClassDict = {}, radius_list = []
center_list = []
for i ∈ length(df) do
    p_i = df[i] ∈ G
    p_{i±n} = df[i ± n] ∈ T                ▷ n は任意の値
    t_{±n} = TimeDiff(p_i, p_{i±n})
    r_{i±n} = ComputeRadius(t_{±n}, V_tourist)
    radius_list.append(r_{i±n})
    center_list.append(p_i)
    PoiClassDict[poiclass] = radius_list, center_list
end for
ステップ 3:POI クラスごとのヒートマップを構築
Gridmap_dict = BuildGridmap(PoiClassDict)
for gridmap ∈ Gridmap_dict do
抽出したエリアの凸包を計算したのちに，重心を計算
    Centeroid = ComputeCenteroid(gridmap)
計算した重心の値と正解位置のユークリッド距離を計算
    Result = EuclideanDistance(Centeroid, GroundTruth)
    ResultTable.add(Result)
end for
return ResultTable
```

1. **ユーザごとの速度を計算し，観光客の平均歩行速度を計算:Algorythm2，2-10 行目**
   - **ComputeVelocity** を用いて，投稿データフレームに含まれる各ユーザの連続投稿から移動距離と移動時間を求め，平均歩行速度を算出.
   - ユーザごとの平均歩行速度の歩行速度の平均値を「歩行観光客の平均歩行速度」として算出.
2. **POI クラスごとの円情報を計算:Algorythm2，11-22 行目**
   - **TimeDiff,ComputeRadius** により，連続する「ジオコーディングにより得られた投稿」の間に位置する「未確定投稿」に対して，円の中心座標と半径を算出する.
   - 得られた (中心座標，半径) を POI クラス *poiclass* ごとに *PoiClassDict* という辞書形式で保存.
3. **POI クラスごとのグリッドマップを構築:Algorythm2，23-25 行目**
   - **BuildGridmap** を用いて，*PoiClassDict* 内のすべての (中心座標, 半径) を poiclass ごとに指定範囲のグリッド *gridmap* を構築.
   - gridmap の構築時には手法 1 と同様の重み（count, area）を用いて各セルの値を更新.
   - POI クラス別に二次元配列 *gridmap* を得て，*Gridmap_dict* として格納.
4. **凸包や重心の抽出:Algorythm2，26-31 行目**
   - グリッドマップの各セルから最も値の高いセルを特定し，そのセルを全て内包するような凸包を算出.（**ComputeCenteroid**）
   - 凸包の重心を算出し，重心と正解位置とのユークリッド距離を計算.（**EuclideanDistance**）

4. 実験

ここまでで提案した手法を評価するために，実際の観光地である，奈良県奈良市のならまちエリア（詳しい範囲は後述）における観光実験を行った．本章では実験の概要について示す.

4.1. 実験概要

本実験は提案手法の有効性と中谷ら [10] の提案しているジオコーディングモデルの結果をアンカーポイントとした条件下での手法 1，2 の精度について検証することを目的として行った．2024 年 6 月 1 日，2 日にわたって，奈良先端大の学生 16 人を募り，奈良県奈良市のならまちエリアの奈良女子大，東向商店街，元興寺，春日大社，東大寺二月堂を囲うようなエリアにおいて，松田ら [9] が開発したソーシャルメディアであるレポっとを用いて観光しながら目についたものを投稿してもらいデータ収集を行なった．本実験により得られた投稿数は 1617 件であった.

4.2. 投稿要件

本実験を行うにあたり，提案手法を適用するのに十分な投稿数，ジオコーディング可能な文字数の確保．そして投稿内容の性質がジオコーディング可能なものとそうでないものどちらかに偏りすぎないように，以下の投稿要件と投稿対象 PoI の例を示し実験参加者に説明した．投稿要件と投稿対象の例を以下の図 2， 3 に示す.

図 2　“広く一般に知られているもの” の例

図 3　“一般に知られていない珍しいもの” の例

- **投稿件数**：100 件以上（1 人当たり）
- **投稿文字数**：45 文字程度
- **投稿対象 (以下ののような PoI を 50% ずつの割合で投稿することが望ましい)**
  1. 対象が広く一般に知られているもの（例，東大寺の大仏，春日大社など）
  2. 一般に知られていない珍しいもの（例，ご当地マンホールなど）

4.3.　結果・考察

本実験では計 1617 件の投稿が得られた．そのうち，ジオコーディングモデルによりジオコーディングされた投稿は 829 件であった．

本研究で得られたデータのうち，同じ PoI について投稿されているものにラベルづけを行うためにアノテーションツールを用いて全ての投稿に対して PoI のラベリングを行い，941 件の推定対象 PoI（TPoI）を同定した．そのうち，複数ユーザが言及しているものは 229 件であった．

4.3.1.　手法 1（アンカーポイントの緯度経度としてレポっとに付与されているデータを用いたもの)

本節ではアンカーポイントの緯度経度が 100% 正しくジオコーディングされたという仮定のもとで手法 1 を適用した結果を示す．表 1 はアノテーションツールによって同定された TPoI の重みごとの位置推定結果である．アルゴリズム 1 では，アンカーポイントに理想的な GPS データを用いた場合．重みを「カウント」とした手法が最も高い精度を示し，中央値が 37.00m，平均値が 56.58m であった．この結果から，外れ値に対して頑健であり，投稿数が多い環境で特に有効な手法であることが確認された．半径および面積を重みとした場合に精度が低くなるのは使用しているアンカーポイントの GPS 精度の誤差に影響される可能性が高い．

表 1　推定エリアの凸包の重心から正解位置の距離の平均値，中央値，標準偏差（単位：m）

重み	平均値	中央値	標準偏差	四分位範囲
カウント	**56.58**	**37.00**	62.51	53.03
面積	67.60	40.51	88.45	60.85

4.3.2.　手法 1（アンカーポイントの緯度経度としてジオコーディングモデルの出力結果上位 1 件の場所を用いたもの)

本節では，アンカーポイントとしてジオコーディングモデルの推定結果（上位 1 件）を採用した場合にアルゴリズム 1 を適用した結果について述べる．表 2 に示されている通り，重みとして「面積」を使用した場合が中央値 145.21m，平均値 205.53m と，最も精度の高い結果を示した．一方で，重みを「カウント」や「半径」とした場合には，いずれも中央値や平均値がやや高く，標準偏差および四分位範囲が広い結果となった．これらの結果は，アンカーポイントの精度が完全には保証されていない環境では，円の信頼度を面積に基づいて調整する手法が有効であることを示唆している．

ただし，面積を重みとする手法は，標準偏差が大きく，外れ値が依然として目立つことから見受けられる．これは，アンカーポイントにおけるジオコーディングモデルの誤差が累積し，円の位置や大きさ，信頼度に影響を与えたためと考えられる．

閾値を高く設定した場合アンカーポイントの信頼性が向上するが，その分，利用可能なアンカーポイントの数が減少し，推定精度に悪影響を及ぼす可能性がある．一方で，閾値を低く設定すると，低精度のアンカーポイントが含まれることで外れ値の影響が増加する一方，円の数が増えることで精度が改善する場合もある．これらのトレードオフを考慮し，最適な閾値を設定することが重要である．

4.3.3.　手法 2(アンカーポイントの高精度 GPS を想定した場合)

まず，ジオコーディング結果が 100% 正確に得られると仮定し，GPS 精度が 50m 以下のすべてのポイントをアンカーポイントとして用いた結果を表 3 に示す．重みを「面積」とした場合，中央値が 82.47m，平均値が 134.64m と最も高い精度を示した．さらに，四分位範囲（IQR）が 101.11m と相対的に狭く，推定値が集中している様子がうかがえる．一方，標準偏差は 180.54m と大きく，外れ値が一定数含まれていることも示唆される．

表 2　推定位置と正解位置の距離の平均値，中央値，標準偏差，四分位範囲（単位：m）

重み	平均値	中央値	標準偏差	四分位範囲
カウント	222.99	145.21	205.09	206.16
面積	**205.53**	**145.21**	198.62	171.07

表 4　アンカーポイントの緯度経度に GPS データを用いた場合（GivenPoI 集合）の推定エリアの凸包重心から正解位置の距離（単位：m）

重み	平均値	中央値	標準偏差	四分位範囲
カウント	160.05	101.53	182.09	152.88
面積	**156.40**	**100.00**	178.63	147.26

表 5　アンカーポイントの緯度経度にジオコーディングの結果を用いた場合（FilteredGeocoding 集合）の推定エリアの凸包重心から正解位置の距離（単位：m）

重み	平均値	中央値	標準偏差	四分位範囲
カウント	**401.18**	263.32	366.54	475.75
面積	420.14	**215.80**	434.96	497.42

表 3　推定エリアの凸包の重心から正解位置の距離の平均値，中央値，標準偏差，四分位範囲（単位：m）

重み	平均値	中央値	標準偏差	四分位範囲
カウント	139.59	89.70	162.30	118.14
面積	**134.64**	**82.47**	180.54	101.11

このことから，アンカーポイント同士の円の重なりの「面積」が大きいほど位置推定への寄与度を高く設定する手法は，多くのアンカーポイントが円として重なる領域を強調でき，より確度の高い推定が期待できると考えられる．しかし，アンカーポイント自体に外れ値が含まれる場合には，その影響が大きく出る可能性も示唆される．

4.3.4.　アンカーポイントの信頼性と閾値の影響

次に，アンカーポイントの緯度経度の信頼性が推定精度に及ぼす影響を検証するため，GPS 精度の閾値を変化させた場合の推定結果を比較した．具体的には，ジオコーディング可能な PoI の集合である GPoI 集合のうち，ある閾値以下の GPS 精度をもつポイントの元 GPS 情報をアンカーポイント（GivenPoI 集合）とした場合と，それらのポイントをジオコーディングモデルにより推定した緯度経度（FilteredGeocoding 集合）をアンカーポイントとした場合で推定精度を比較する．ここでは閾値を 50m に設定した例を表 4 および表 5 に示す．

GivenPoI 集合（表 4）では，面積重みの中央値が 100.00m

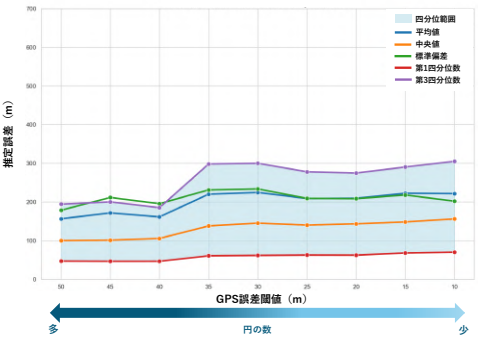

図 4　アンカーポイントの緯度経度に GPS データを用いた場合の各統計量の変化

で，PoI 推定に十分活用できるレベルの精度が得られている．これは元の GPS データが高い信頼度をもつ場合，円同士の重なり面積を重視する手法が有効に働くことを示唆している．一方，FilteredGeocoding 集合（表 5）では，面積重みの中央値が 215.80m と大きく，ジオコーディングに伴う推定誤差が位置推定精度に大きく影響していることが分かる．

さらに，GPS 精度の閾値を変動させたときの推定誤差の統計量の変化を図 4 および図 5 に示す．理想的な GPS データ（GivenPoI）を用いる場合（図 4）には，閾値を緩和するとアンカーポイント数が増加し，推定精度の平均値・中央値が改善するとともに，四分位範囲も狭まる傾向が確認できる．これは，アンカーポイント（円）の総数が増えることで，それらの重なり領域をより適切に抽出できるためと考えられる．

一方，ジオコーディングモデルの結果を用いた場合（図 5）では，GPS 精度の閾値を緩和すると誤差の大きい推定値が混在しやすくなり，平均誤差が悪化するとともに，四分位範囲が広がる傾向がみられる．しかし，中央値の変化は比較的小さく，ある程度の範囲内での外れ値に対して本手法が頑健な一面も確認できる．

以上の結果から，以下の知見が得られる．

1. アンカーポイントの信頼性が高い（GPS 精度が高い）場合には，円同士の重なり面積を重みに組み込むことで推定精度を向上できる．これは推定結果の中央値や四分位範囲の改善が顕著であり，PoI 推定の有効性が示唆される．
2. 一方で，外れ値が含まれた場合やジオコーディングモデルによる推定誤差が大きい場合には，面積重みを用いることでかえって大きな外れ値を生む可能性がある．すなわち，ジオコーディング結果の誤差が大きいアンカーポイントを多数含む状況では，推定精度のばらつきが増加し，平均誤差が悪化することが分かった．

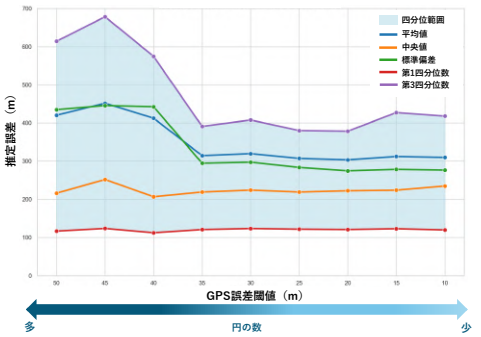

図 5　アンカーポイントの緯度経度にジオコーディングの結果を用いた場合の各統計量の変化

5. 結論

本研究では，既存の PoI 位置推定では位置推定することのできない GoogleMap や OpenStreetMap といった地図 DB に登録されていない未登録の PoI の位置推定を行うことを目的として，観光地での観光客の時系列ソーシャルメディア投稿を用いた位置推定フレームワークを提案した．提案フレームワークの検証のために，奈良県奈良市において実験を行った．奈良県ならまちエリアを対象として，被験者 16 人で約 1600 の投稿データに対して，提案手法 1，2 を適用し比較検討をおこなった．その結果，手アンカーポイントとして得られる緯度経度がジオコーディングモデルによって 100% の精度で得られたという仮定のもとで手法 1 を適用した際の推定誤差は 37m（中央値），アンカーポイントにジオコーディングモデルの推定結果を用いた状態で手法 1 を適用した際の未登録 PoI の推定誤差は 145.2m（中央値）であった．また，手法 2 においてアンカーポイントを緯度経度がジオコーディングモデルによって 100% の精度で得られたという仮定のもとでの推定誤差は 82.5m（中央値），ジオコーディングモデルが推定したポイントをアンカーポイントとした場合の推定誤差は 215.8m（中央値）であった．これらの結果は，本提案手法は位置推定対象が地図データベースに登録されているかどうかに関わらず，既存手法と同等の位置推定精度を示しており，地図データベースに未登録の PoI を推定する手法として有効であることが示された．

今後の展望として，本提案手法で行われた位置推定に対して不確実性も考慮するために推定出力を混合確率分布とすることが考えられる．提案手法の円を混合確率分布とすることで，円を描く範囲にどの程度の確率で推定対象のポイントが含まれているかを確率的に表すことができると考えられる．また，現状はジオコーディングモデルの上位 1 件を推定ポイントとして使用しているが，使用しているジオコーディングモデルは日本語旅行記データセットにおいて，上位 10 件のジオコーディングで 87.7% の精度を達成しており，これら全てのジオコーディングポイントを使用することで，より多くの円を描くことができそれによりさらなる推定精度の向上が期待できると考えられる．

参考文献

[1] 福井一喜：東京大都市圏に居住する若者の観光・レジャーにおける SNS 利用―「SNS 映え」を超克する若者たち―，*E-journal GEO*, Vol. 14, No. 1, pp. 1–13（オンライン），10.4157/ejgeo.14.1 (2019).

[2] Hulden, M., Silfverberg, M. and Francom, J.: Kernel Density Estimation for Text-Based Geolocation, *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 29, No. 1 (online), 10.1609/aaai.v29i1.9149 (2015).

[3] Roller, S., Speriosu, M., Rallapalli, S., Wing, B. and Baldridge, J.: Supervised Text-based Geolocation Using Language Models on an Adaptive Grid, *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, Association for Computational Linguistics, pp. 1500–1510 (2012).

[4] Wing, B. and Baldridge, J.: Hierarchical Discriminative Classification for Text-Based Geolocation, *Conference on Empirical Methods in Natural Language Processing* (2014).

[5] Li, M., Lim, K. H., Guo, T. and Liu, J.: A Transformer-Based Framework for POI-Level Social Post Geolocation, *Advances in Information Retrieval*, Springer Nature Switzerland, pp. 588–604 (2023).

[6] Radford, B. J.: Regressing Location on Text for Probabilistic Geocoding, *Proceedings of the 4th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE 2021)*, Association for Computational Linguistics, pp. 53–57 (2021).

[7] Iso, H., Wakamiya, S. and Aramaki, E.: Density Estimation for Geolocation via Convolutional Mixture Density Network, (online), 10.48550/arXiv.1705.02750 (2017).

[8] Dai, R., Luo, J., Luo, X., Mo, L., Ma, W. and Zhou, F.: Multi-modal Representation Learning for Social Post Location Inference, *ICC 2023 - IEEE International Conference on Communications*, pp. 6331–6336 (online), 10.1109/ICC45041.2023.10279649 (2023).

[9] 松田裕貴，河中祥吾：Web ブラウザ上で動作する市民参加型写真収集アプリの開発と運用，第 27 回社会情報システム学シンポジウム（ISS27），pp. 1–5 (2021).

[10] Nakatani, H., Teranishi, H., Higashiyama, S., Sawada, Y., Ouchi, H. and Watanabe, T.: A Text Embedding Model with Contrastive Example Mining for Point-of-Interest Geocoding, *Proceedings of the 31st International Conference on Computational Linguistics*, Association for Computational Linguistics, pp. 7279–7291 (2025).