卒業論文　　2010 年度 (平成 22 年度)

GLoBES：位置情報を用いた グループ検出アルゴリズムの設計と実装

指導教員

慶應義塾大学 環境情報学部
徳田 英幸
村井 純
楠本 博之
中村 修
高汐 一紀
重近 範行
Rodney Van Meter
植原 啓介
三次 仁
中澤 仁
武田 圭史

慶應義塾大学 総合政策学部
蛭田 慎也

hiru@ht.sfc.keio.ac.jp

卒業論文要旨　　2010 年度 (平成 22 年度)

GLoBES：位置情報を用いた グループ検出アルゴリズムの設計と実装

論文要旨

本論文では，ソーシャルメディアにおける位置情報付きの発言を解析することで，ユーザ間の類似度を推定する手法を提案する．近年，ソーシャルメディアの普及により，誰もが手軽に情報を発信，共有できるようになった．また，GPS などの位置情報取得機能付きの携帯電話やスマートフォンの普及も進んでおり，ユーザの正確な位置情報も手軽に取得できるようになった．このような背景により，ソーシャルネットワークにおける位置情報の共有アプリケーションが急速に普及し始めている．ソーシャルネットワークではユーザの交友関係が大きな価値を生むため，ユーザ同士の関係や趣味などが似ているユーザを発見する手法が求められる．このような類似ユーザの推定においては，多くの研究が行われている．

ソーシャルネットワークにおける類似ユーザを推定する手法はいくつか提案されている．第一に既存のユーザ同士のつながりであるソーシャルグラフの解析を行う方法がある．この手法では簡易的に類似ユーザの推定を行うことができる．しかし，オンライン上のつながりしか参照しないため，実世界における行動パターンや興味を持つ場所が似ているユーザを発見することはできない．第二に，GPS ロガーなどの端末を持たせ，位置情報の軌跡から類似ユーザを推定する手法が研究されている．この手法では実世界における行動の類似性を反映することができるが，端末のバッテリを多く消費したり，プライバシが考慮されていないという問題があるため，実際のユーザに対して適用することは困難である．

そこで本研究では，これらの課題を解決するため，ソーシャルメディアにおける位置情報付きの発言を利用した類似ユーザの推定手法 GLoBES (Grouping Algorithm Based on Location of Micro-Blog Entries) を提案する．本研究では，発言に位置情報を付加可能なソーシャルメディアを対象とする．GLoBES は，発言取得モジュール，発言解析モジュール，クラスタリングモジュール，類似度計算モジュール，グループ情報取得モジュールで構成される．これらを Linux サーバ上に実装し，各モジュールの評価を行った．評価は定量的評価として，各モジュールの評価を行い，本システムの推定した類似ユーザ情報の妥当性を明らかにする．また，定性的評価として既存研究と GLoBES の比較を行う．

本研究は，既存の類似ユーザ推定手法が用いていた独自ネットワークの構築や特殊な端末を必要としないため，既存のソーシャルネットワークにおける全世界のユーザ情報を活用することが可能になる．また，推定した類似ユーザ情報は再び既存のソーシャルネットワークに対して適用可能であるという点でも有益である．

キーワード：

1 位置情報サービス　2 ソーシャルメディア　3 類似ユーザ
4 データマイニング　5 プライバシ

慶應義塾大学 総合政策学部総合政策学科
蛭田 慎也

Abstract of Bachelor's Thesis

Academic Year 2010

GLoBES: A Spatio-Temporal Grouping Algorithm using Micro-Blog Entries

Summary

In this thesis, we propose a method to measure similarities among users by mining location of micro-blog entries.

Recently, social media such as Twitter and Facebook attract public attention and many users can now easily share the information. Furthermore, spread of devices having location acquisition technologies (GPS, Wi-Fi measurement, etc.) makes it possible to acquire coordinates at many places. From such a background, location sharing applications (LSA) on social networking service (SNS) become widely accepted. On the social networking service, user's relationship has important value. Therefore, many researchers are working on the issue to mining user's similarity.

Mining social graph on SNS is an existing method to measure similarities among users. However, this method can not approach the user's locational similarity because it refers only on online network among users. Mining people's mobility pattern by GPS logger device is an another existing method. But this is not applicable to the general purposes because such devices have limited use of battery and acquiring location at everywhere ignores user's privacy.

In order to solve this issue, We propose GLoBES (Grouping Algorithm Based on Location of Micro-Blog Entries) to measure similarities among users by mining location of micro-blog entries. In our research, we measure similarities among users by micro-blog entries, and then build an API to send measured similarities to LSA. GLoBES is designed to work on a server to measure similarities and communicate with API. In order to evaluate GLoBES, we compare the validity of the measured similarity and actual user's relationship.

Keyword :

1 Location Based Service 2 Social Media 3 Similar User 4 Data Mining 5 Privacy

Shinya Hiruta
Faculty of Policy Management
Keio University

目 次

図 目 次

表 目 次

第1章　序論

本章では，研究動機について述べ，本研究の目的を提示する．後半では論文全体の構成についてまとめる．

1.1 研究動機

近年，位置情報共有アプリケーションが普及している．位置情報共有アプリケーションとは，ユーザの所持している端末から位置情報を取得し，その情報をユーザ同士で共有したり，ユーザにサービスを提供したりするアプリケーションである．ユーザの現在位置周辺の情報を検索したり，目的地までの道のりを案内してくれるナビゲーションサービス，位置情報取得機能付きの端末を持たせた人や乗り物，登録した友人の位置情報などを検索する第三者検索，GPS 情報によりユーザが実際に足を運んだ場所に関連してゲームが進行していく位置情報を用いたゲームなどがその一例である．このような位置情報共有アプリケーションが普及した背景の一つには，位置情報取得機能付きの携帯電話や，iPhone[1] や Android[20] などのスマートフォンの普及がある．このようなスマートフォンを利用した位置情報共有アプリケーションは，今後ますます一般的になってくると考えられる．

また，近年 Facebook[6] や Twitter[29] などのソーシャルメディアの利用者も急速に増加している．これらのソーシャルメディアの流行により，情報を公開し，共有するという行為をユーザがより気軽に行うことができるようになった．このような傾向は，位置情報共有アプリケーションにも大きな影響を与えている．近年の位置情報共有アプリケーションの多くが，ネットワーク上での人と人とのつながりを重視し，位置情報を他人と共有する行為に重点を置いている．よって，今後の位置情報共有アプリケーションは，ソーシャルメディアとの連携がより重要になってくると考える．

ソーシャルメディアにおけるオンライン上でのつながりの数と，ユーザが訪れた物理的な位置には正の相関関係があるという既存研究がある [4]．実世界でのユーザの位置情報を取得して行動パターンや趣味が似ているユーザを見つけ出し，オンラインでのソーシャルネットワークへフィードバックすることができれば，同じ地域に住むユーザのグループに対して地域限定の広告やクーポンを配信するなどの新たなサービスの可能性が生まれると考えられる．このような，行動パターンや興味を持つ場所が似ているユーザのことを，本論文では「類似ユーザ」と定義する．実世界でのユーザの行動から，オンライン上での類似ユーザを推定する手法が，今後ますます求められるようになると考える．

関連研究として，ユーザの位置情報から類似するユーザを推定する研究は数多く行われている．これらの研究の多くは，ユーザの端末から GPS ログを定期的に取得し，移動軌跡のログを解析するという手法を用いている．しかし，このような手法は，バッテリーが限られる携帯端末においては現実的ではない．さらに，ユーザの現在位置情報を常に取得してしまうのはプライバシの問題がある．

このような問題を解決するため，本研究では Twitter などのソーシャルメディアにユーザが自ら公開した位置情報を用いて類似ユーザを推定する手法を考案する．従来の手法では，連続した膨大な GPS ログからユーザが立ち寄った場所や興味を持った場所を推定する必要があった．しかし，ソーシャルメディアにおいてはユーザが自ら興味を持った場所の位置情報や感想などを発言している．このような発言情報を用いれば，移動軌跡のみを用いる手法に比べてユーザの行動パターンや興味を正確に推定できると考えられる．また，ユーザの発言時のみに位置情報を取得するためプライバシを侵害するおそれが少なく，常に位置情報を取得する手法と比べて，端末の限られたバッテリを有効に活用することが可能である．

1.2 研究目的

本研究では，ソーシャルメディアによる位置情報付きの発言から類似ユーザを検出し，位置情報共有アプリケーションを支援することが目的である．現状でも，ソーシャルネットワーク上におけるつながりの関係から，類似しているユーザを推定してくれるサービスなどは多く行われている．しかし，これらはあくまでもオンライン上のつながりのみを参照しているため，必ずしも実世界における行動の類似性を反映しているものではない．今後，ユーザが気軽に位置情報を共有するようになった際，それらの情報をもとに，より実世界の交友関係に基づいた類似ユーザを推定できるような手法の研究が必要である．

行動パターンや興味を持つ場所が似ているグループ情報がアプリケーションから利用可能になれば，グループの人数や属性に応じた情報提供や広告配信など，新たなサービスの可能性が生まれると考えられる．このようなサービスを実現するために，本研究では GLoBES を構築し，次世代の位置情報共有アプリケーションの基盤を構築することを目指す．

1.3 本論文の構成

本論文は，第 2 章において本研究の目的と背景について述べ，位置情報共有アプリケーションと位置情報取得機能付き端末についてそれぞれ具体例を挙げ，比較する．第 3 章では，本研究の提案する GLoBES の設計について述べる．第 4 章では，GLoBES の実装について述べる．第 5 章では，GLoBES の各モジュールについて定量的評価を行う．第 6 章では，本研究と関連研究を比較し，本研究の有益な部分を述べる．最後に，本論文のまとめと，本研究の課題と展望について述べる．

第2章　背景

本章では，まず本研究の背景として，位置情報共有アプリケーションの普及と位置情報取得機能付き端末の普及について述べ，本研究の目的である位置情報共有アプリケーションの支援について述べる．次に，既存の位置情報共有アプリケーションとして，具体的な位置情報サービスを挙げ，それぞれのサービスに対する技術的な要求についてまとめる．最後に，位置情報取得技術の発達について述べる．

2.1 本研究の目的

本節では，まず本研究の背景として位置情報共有アプリケーションの普及と位置情報取得機能付き端末の普及について述べる．そして，本研究の目的である位置情報共有アプリケーションの支援について述べる．

2.1.1 研究背景

近年，ナビゲーションサービスや第三者検索，位置情報を用いたゲームなどの，位置情報共有アプリケーションが普及している．このような位置情報共有アプリケーションが普及した背景には，位置情報取得機能付きの携帯電話や，iPhone[1] や Android[20] などのスマートフォンの普及がある．これらの高機能端末の市場規模は国内外共に拡大しており，位置情報共有アプリケーションは今後ますます一般的になってくると考えられる．

また，近年 Facebook[6] や Twitter[29] などのソーシャルメディアの利用者も急速に増加している．それに伴い，位置情報共有アプリケーションの多くが，ソーシャルネットワークにおける他人とのつながりを重視したものへと変化してきている．従って，今後の位置情報共有アプリケーションは，ソーシャルメディアとの連携がより重要になってくると考える．

さらに，ソーシャルメディアにおけるオンライン上でのつながりの数と，ユーザが訪れた物理的な位置には正の相関関係があるという既存研究がある [4]．ユーザの位置情報を 10 分間隔で取得した軌跡データを解析することで，2 ユーザ間の友達関係を推定することを可能にしている．このように，実世界でのユーザの位置情報から行動パターンや趣味が似ているユーザを見つけ出し，オンラインでのソーシャルネットワークへフィードバックすることができれば，新たなアプリケーションの可能性が広がるだろう．したがって，実世界におけるユーザの位置情報から類似したユーザを推定する手法が，今後ますます求められるようになると考える．

図 2.1: iPhone 4

図 2.2: twitter.com

2.1.2 位置情報共有アプリケーションの支援

本論文の提案する GLoBES (Grouping Algorithm Based on Location of Micro-Blog Entries) では，実世界の行動パターンを基にユーザ間における類似度を計算する．そして，あるユーザを基準として考えた時，そのユーザから見た類似度が一定の閾値以上のユーザを「類似

ユーザ」と定義する．また，あるユーザに対する類似ユーザ全体のことを,「グループ」であると定義する．

本研究では，ソーシャルメディアによる位置情報付きの発言から類似ユーザを検出し，位置情報共有アプリケーションを支援することが目的である．現状でも，ソーシャルネットワーク上におけるつながりの関係から，類似しているユーザを推定してくれるサービスなどは多く行われている．しかし，これらはあくまでもオンライン上のつながりのみを参照しているため，必ずしも実世界の交友関係を反映しているものではない．今後，ユーザが気軽に位置情報を共有するようになった際，それらの情報をもとに，より実世界の交友関係に基づいた類似ユーザを推定できるような手法の研究が必要である．

行動パターンや趣味が似ているグループ情報がアプリケーションから利用可能になれば，グループの人数や属性に応じた情報提供や広告配信など，新たなサービスの可能性が生まれると考えられる．このようなサービスを実現するために，本研究では GLoBES を構築し，次世代の位置情報共有アプリケーションの基盤を構築することを目指す．

2.2 位置情報共有アプリケーション

本節では，まず既存の位置情報サービス例を挙げ，それぞれのサービスに対する技術的な要求についてまとめる．次に，位置情報共有アプリケーションを，位置情報を公開する動機，位置情報公開対象の二つの観点から比較する．最後に，知り合い情報を用いたサービス例を挙げ，その課題について述べる．

2.2.1 既存の位置情報サービス

既存の位置情報サービスを，ナビゲーション，第三者検索，位置ゲームの三つに分類する．表 2.1 を踏まえ，それぞれについて具体的なサービスを挙げ，概要について述べる．

ナビゲーション

単純に地図上にユーザの現在位置を表示するだけのサービスから，現在位置周辺の情報検索や，設定した行き先までのナビゲーションをしてくれるサービスまで存在する．モバイル GoogleMaps[10] では，携帯電話やスマートフォンで，現在位置検索や周辺情報検索，ルート検索サービスを提供している．EZ ナビウォーク [13] や駅探 [5] などは，各社携帯電話ユーザ向けに有料サービスを提供している．

これらのナビゲーションサービスにおいては，ユーザの位置情報の取得機能と地図の表示機能などが求められる．さらに，地図上にユーザの周辺情報を表示したり，目的地までのルートを表示する機能も必要とすることもある．主に使用するユーザ本人に対してのみサービスを提供するもので，ソーシャルネットワーキングなどによる他のユーザとのコミュニケーション機能は必要としないものが多い．また，他のユーザと位置情報を共有する要求もない．

第三者検索

第三者検索とは，位置情報取得機能付きの端末を持たせた人や乗り物，登録した友人の位置情報などを検索するサービスである．主に防犯や見守り目的で使用され，携帯電話各

表 2.1: 位置情報サービスに対する技術的な要求

サービス	位置情報の取得	SNS 機能	地図の表示	位置情報の共有
ナビゲーション				
モバイル GoogleMaps[10]	○	×	○	×
EZ ナビウォーク [13]	○	×	○	×
駅探 [5]	○	×	○	×
第三者検索				
イマドコサーチ [19]	○	×	○	△
安心ナビ [12]	○	×	○	△
位置ナビ [24]	○	×	○	△
ココセコム [22]	○	×	○	△
位置ゲーム				
コロニーな生活☆ PLUS[3]	○	○	×	○
ケータイ国盗り合戦 [18]	○	○	×	△
Foursquare[7]	○	○	○	○
はてなココ [11]	○	○	○	○
ロケタッチ [17]	○	○	○	○
Parallel Kingdom[21]	○	△	○	○

社が提供するイマドコサーチ [19]，安心ナビ [12]，位置ナビ [24] や，ココセコム [22] などがある．

第三者検索サービスにおいては，検索を行うユーザと，検索対象となる端末やユーザが物理的に離れた場所にいるという点が，他のサービスと大きく異なる．また，位置情報は特定の相手のみと共有され，不特定多数のユーザに共有されることはない．これらのサービスの多くは，位置情報の共有相手として登録されるユーザは少数であるため，ソーシャルネットワーク機能は持たず，ユーザが手動で共有相手を設定する．

位置ゲーム

位置ゲームとは，携帯電話やスマートフォンの GPS 情報を用いて，ユーザが実際に足を運んだ場所に関連してゲームが進行していくものを指す．育成型ゲームのコロニーな生活☆ PLUS[3]，シナリオ進行型ゲームのケータイ国盗り合戦 [18] などが多くのユーザを集めている．主にスマートフォン向けには，位置情報ベースのソーシャルネットワーキングサービスである Foursquare[7] やはてなココ [11]，ロケタッチ [17] などがある．また，Parallel Kingdom[21] は，GPS 情報を用いた MMORPG (Massively Multiplayer Online Role-Playing Game) である．

位置ゲームにおいては，それぞれのゲームのコンセプトにより技術的な要求も異なってくる．ユーザの移動距離をゲーム内の仮想通貨に変換するために位置情報を用いるもの [3] や，国土を一定のエリアに分割し，エリア単位での移動によるシナリオを楽しむゲーム [18] においては，現在位置を地図に表示することは不要である．一方，ユーザが訪れた場所を

他のユーザと共有するチェックイン型のサービス [7][11][17] では，地図上に他のユーザの位置を表示したり，位置情報を共有する機能が求められる．また，ほとんどの位置ゲームにおいて，他のユーザと交流するソーシャルネットワーク機能は必須である．

2.2.2 位置情報共有アプリケーションの分類

位置情報を公開する動機

従来の位置情報共有アプリケーションは，防犯や見守り目的などで端末の位置情報を検索するサービス [19][12][24][22] や，ナビゲーションを行うサービス [13] が主流であった．これらは，特定の目的があった上で，位置情報を利用するアプリケーションであると分類できる．本論文ではこのようなアプリケーションを，目的重視のアプリケーションと呼ぶ．

一方，Twitter[29] や Facebook[6] などのソーシャルメディアの流行に伴い，ユーザの位置情報を公開し，共有することに重点を置いたサービス [7][11][17] が普及し始めている．これらのサービスは，位置情報を公開すること自体が目的の一つとなっている点で，従来の位置情報共有アプリケーションとは大きく異なっている．既存研究によると，ユーザが位置情報を公開する動機は，位置情報を共有することによりソーシャルネットワーク上でのつながりを強化し，ソーシャルキャピタルを築くことであると分析されている [26]．本論文ではこのようなアプリケーションを，社会性重視のアプリケーションと呼ぶ．

位置情報公開対象

目的重視のアプリケーションでは，1 対 1，1 対数人にしか位置情報を公開しないものが一般的であった．例えば，見守りサービスにおいて端末の位置情報を取得することができるのは保護者など限られた人物のみである．

一方，社会性重視のアプリケーションでは，1 対複数，1 対全員に位置情報を公開するものが多い．Foursquare[7] などの位置ゲームでは，ユーザが訪れた場所の位置情報を，サービスを利用している人全員に対して公開し，共有している．

2.2.3 類似ユーザ情報を用いたサービス

ソーシャルメディアや位置ゲームなどの社会性重視のアプリケーションにおいては，知り合いを登録できる機能がほぼ必ず実装されている．このような知り合い関係の情報を用いたサービスとして，図 2.3 に，Foursquare における友達のチェックイン一覧表示を示す．また，ソーシャルメディアでは，既存のユーザ間のつながりから知り合いである可能性の高いユーザを提示してくれる機能も提供されている．図 2.4 に，Facebook の「知り合いを検索」機能による提示結果を示す．

しかし，これらはあくまでもオンライン上のつながりのみを参照しているため，必ずしも実世界の行動パターンが似ているユーザを提示してくれるわけではない．今後，多くのユーザが位置情報取得機能付き端末を持ち歩くようになると，実世界の位置情報を用いて，行動パターンや興味を持つ場所が似ているユーザを推定する技術が必要となる．

図 2.3: Foursquare 友達の最近のチェックイン一覧

図 2.4: Facebook 「知り合いを検索」機能

2.3 位置情報取得技術の発達

本節では，まず位置情報取得機能付き端末の普及について述べる．次に，位置情報取得技術を 4 種類に分類し，位置情報共有アプリケーションにおける有用性について述べる．

2.3.1 位置情報取得機能付き端末の普及

近年，iPhone[1] や Android[20] などのスマートフォンと呼ばれる高機能携帯電話が急速に普及し始めている．2008 年現在，世界市場においてスマートフォンの出荷台数は 1 億 3000 万台を突破し，国内市場においても 158 万台を記録した．国内外共に市場規模は拡大しており，2012 年には世界市場では 2 億 3000 万台，国内市場では 360 万台に達すると予想されている [31].

これらのスマートフォンでは，GPS，SkyHook[23]，PlaceEngine[14] などの技術を用いて，緯度経度の座標や現在位置情報を取得することができる．位置情報共有アプリケーションにおいては，ユーザの位置情報を取得することが必須である．ユーザが普段持ち歩く携帯電話のようなデバイスに位置情報取得機能が搭載されることで，今後位置情報共有アプリケーションはより一般的になると考えられる．

2.3.2 位置情報取得技術の分類

GPS ロガーを用いた方法

GPS ロガーを用いた方法について述べる．図 2.5 と図 2.6 は代表的な GPS ロガーである．GPS ロガーは，主にバッテリで駆動し，GPS から測位した位置情報を定期的に端末内のメモリなど記録するものである．GPS ロガーを用いると，細かい粒度の位置情報を長期間取得することが可能になる．しかし，実用的な位置情報共有アプリケーションにおいては，ユーザに常に GPS ロガーを携帯させることは困難である上，GPS データを端末から回収する手間も大きいため，効果的な手法とは言えない．

図 2.5: GlobalSat DG-200

図 2.6: Magellan eXplorist 710

携帯電話や PHS 基地局情報を用いた手法

携帯電話や PHS 基地局情報を用いた手法について述べる．一般的に，携帯電話は一つの基地局で数百 m〜数 km のエリアをカバーするマクロセル方式が採用されており，PHS では基地局あたり数十 m〜数百 m のエリアをカバーするマイクロセル方式が採用されている [30]．これらの基地局情報を利用することで，ユーザの現在位置を特定することが可能になる．基地局情報を用いる利点としては，GPS 受信モジュールなどが搭載されていない低機能な端末でも位置情報を取得できるという点である．しかし，取得できる位置情報の粒度は基地局の設置されている間隔に依存するため，数十 m〜数 km の誤差が発生する．そのため，ユーザの大まかな位置を取得するには有用だが，立ち寄っている建物や移動ルートを取得するためには精度が不足してしまうという欠点がある．

携帯電話搭載の GPS 情報を用いた手法

携帯電話搭載の GPS 情報を用いた手法について述べる．2007 年 4 月に施行された事業用電気通信設備規則の改正により，施行後に発売される携帯電話には GPS モジュールが搭載されることが義務づけられている [25]．そのため，多くの携帯電話で GPS による位置情報取得を行うことが可能である．GPS 単独による測位では 10m 程度の誤差が発生するが，ユーザが現在立ち寄っている建物を判別する程度の精度は十分確保できる．従って，位置情報共有サービスを構築する上で，携帯電話搭載の GPS を利用することが現状では最も現実的だと考えられる．しかし，屋内や地下など GPS の電波が受信できない場所では，現在位置を特定できなくなってしまう欠点がある．また，GPS モジュールの動作は比較的大きな電力を消費するため，頻繁に GPS を受信するとすぐにバッテリを消耗してしまうという欠点もある．

無線 LAN 基地局情報を用いた手法

無線 LAN 基地局情報を用いた手法について述べる．無線 LAN 接続に対応した携帯端末においては，Skyhook[23] や PlaceEngine[14] といった無線 LAN 基地局情報を利用した測位が可能である．これらは，近くの無線 LAN アクセスポイントの MAC アドレスを取得し，サーバに問い合わせることで，数十 m 程度の精度の位置情報を取得できるものである．利点としては，GPS の弱点であった，屋内や地下など GPS の電波が届かない場所においても現在位置を測位することが可能な点である．しかし，無線 LAN 接続に対応していない携

帯電話や，周りに無線 LAN アクセスポイントが無い状況では利用できないため，GPS など他の測位技術と相補的に利用することが効果的だと考えられる．

2.4 本章のまとめ

本章では，本研究の背景と目的について述べ，既存の位置情報共有アプリケーションと位置情報取得技術について比較，分類した．

本研究の目的の節では，本研究の背景として位置情報共有アプリケーションの普及と位置情報取得機能付き端末の普及についてまとめ，本研究の目的である位置情報共有アプリケーションの支援について述べた．

位置情報共有アプリケーションの節では，既存の位置情報サービスをナビゲーション，第三者検索，位置ゲームの 3 つに分類し，それぞれにおける技術的な要求についてまとめた．次に，位置情報共有アプリケーションを位置情報を共有する動機，位置情報公開対象の二つの観点から比較し，実世界の行動パターンや興味に基づく類似ユーザ推定技術の必要性について述べた．

位置情報取得技術の発達の節では，スマートフォンの普及と，それらの端末が持つ位置情報取得機能についてまとめた．次に，位置情報取得技術を分類し，それぞれの利点と欠点について述べた．

第3章　GLoBESの設計

本章では，ソーシャルメディアに投稿された位置情報付きの発言を収集することによりユーザ間の類似度を推定するGLoBES (Grouping Algorithm Based on Location of Micro-Blog Entries) の設計について述べる．

まずGLoBESの概要について述べ，本研究の想定環境についてまとめる．さらに，GLoBESの設計とモジュール構成について述べる．

3.1 概要

位置情報取得機能付き端末の普及が進み，誰でもどこでも位置情報付きの発言を共有できるようになった環境においては，実世界の行動パターンや興味が似ているユーザを推定する手法が求められる．そこで，本研究ではソーシャルメディアに投稿された位置情報付きの発言を収集することによりユーザ間の類似度を推定する GLoBES（Grouping Algorithm Based on Location of Micro-Blog Entries）を構築する．GLoBES の構築により，アプリケーションからの要求に応じて類似ユーザ情報を提供することで，実世界の行動パターンなどが似ているグループへの広告配信など，新たなサービスの基盤を築くことを目指す．

GLoBES は，実世界の行動パターンや興味を持つ場所が似ているユーザを検出し，各ユーザの類似度を算出する．ユーザ間の類似度が閾値以上のユーザを，類似ユーザと定義する．また，あるユーザを基準にした際，ユーザから見た類似ユーザをグループとして取得し，その情報をアプリケーションから利用可能にする．

図 3.1 に GLoBES の動作概要を示す．まずソーシャルメディアから Taro, Hanako, Okina の三人の位置情報付き発言を取得する．次に，すべてのユーザの組み合わせで類似度を計算する．ここでは，Taro から Hanako に対する類似度が 0.9 とし，Taro から Okina に対する類似度が 0.1 だと仮定する．また，類似度が 0.5 を超えた場合に類似ユーザと判断するという閾値を設定する．最後に，Taro を基準とした場合，類似度が閾値の 0.5 以上である Hanako をグループとして判定する．アプリケーションからの要求に応じて，判定したグループ情報を提供する．

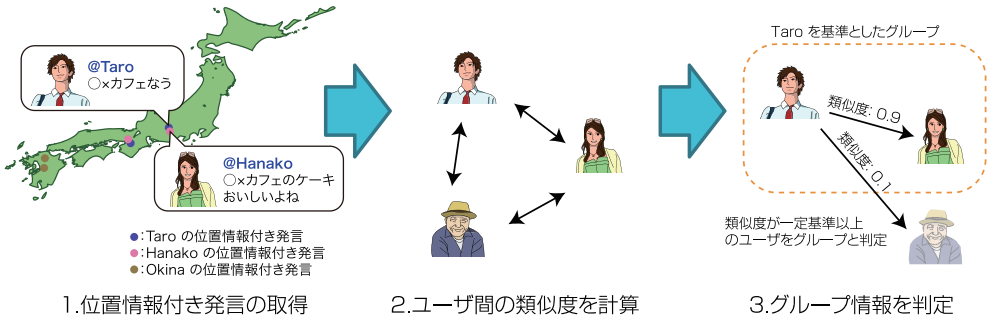

図 3.1: GLoBES の概要

3.2 想定環境

本節では本研究の想定環境として，以下の要素を挙げる．

情報取得対象

ユーザが短い文字列で発言を投稿する，マイクロブログと呼ばれるソーシャルメディアを対象とする．さらに，発言時にユーザの位置情報を付加できることを必要条件とする．

グループ情報の利用対象者

位置情報共有アプリケーションの開発者を対象とする．GLoBES が判定したグループ

情報は，直接エンドユーザに提供されるものではない．位置情報共有アプリケーションを拡張するために，GLoBES が提供するグループ情報を利用することを想定する．

3.3 グループ判定アルゴリズム

本節では，ソーシャルメディアの発言を取得してからグループを判定するまでのアルゴリズムを，順を追って述べる．まず，ソーシャルメディアにおける位置情報付き発言の取得について述べ，位置情報を基に発言をクラスタリングする手法について述べる．次に，クラスタリングを行った発言からユーザ間の類似度を判定する手法について述べる．最後に，計算した類似度情報を基に，グループを判定する手法について述べる．

3.3.1 ソーシャルメディアにおける位置情報付き発言の取得

まず，ソーシャルメディアから位置情報付きの発言を取得する．取得において必要な項目を表 3.1 にまとめた．これらの項目が取得可能なソーシャルメディアであれば，具体的なサービスを問わず適用可能である．

表 3.1: ユーザ類似度判定に必要な項目

項目	形式	必須/任意
発言時刻	年/月/日 時:分:秒	必須
発言ユーザ ID	数値	必須
発言位置情報	緯度, 経度	必須
発言内容	文字列	任意

3.3.2 発言のクラスタリング

取得した位置情報付きの発言を，位置情報を基にクラスタリングを行う．クラスタリングを行う理由は，膨大な発言の中から，位置情報が近い組み合わせを効率的に取得するためである．クラスタリングを行わない場合，すべての発言の組み合わせに対して距離を計算する必要があり，膨大な計算量となるためアルゴリズムがスケールしなくなってしまうという問題がある．

クラスタリングには，既存の複数のアルゴリズムが存在する．表 3.2 に，アルゴリズム名とそれぞれの特徴をまとめた．今回は，分類対象となる発言数が非常に多いため，計算量の少ない K-means 法を利用した．

図 3.2 では，事前実験で取得した発言の位置情報をプロットした結果を示す．Twitter における日本周辺の位置情報付き発言を 2010 年 12 月 4 日から 2010 年 12 月 10 日の 7 日間取得したもので，件数は約 10 万件である．日本列島のほとんどの場所において位置情報付きの発言がされていることが分かる．なお，図の右下に，何も無い海上に位置情報が設定された発言が記録されている．これは，ランダムに位置情報を変えながら発言を繰り返す BOT や，海上を震源とする地震の位置情報を発言する BOT などによるものであった．このようなノイズとなる発言は，全体の 0.3%ほど観測された．

表 3.2: クラスタリングのアルゴリズム比較

分類	アルゴリズム	計算量
階層的	最短距離法・最長距離法・群平均法・ウォード法など	$O(N^2)$
非階層的	K-means 法	$O(Nk)$

図 3.3 では，事前実験のデータをクラスタ数 $K = 10$ でクラスタリングを行った結果を示す．色の境目がクラスタの境界であり，すべての発言は合計で 10 個のクラスタに分割されている．同じクラスタに属する発言は，行動範囲や興味を持つ場所が似ている発言であると解釈する．あるユーザを基準に考えたとき，まずはどのクラスタで何回発言しているかを計算する．そして，他のユーザの中からクラスタを共有する発言が多いユーザほど，基準となるユーザからの類似度が高いと考えられる．

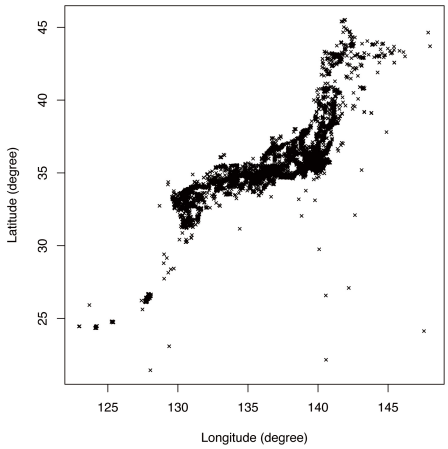

図 3.2: 発言の位置情報をプロットした結果

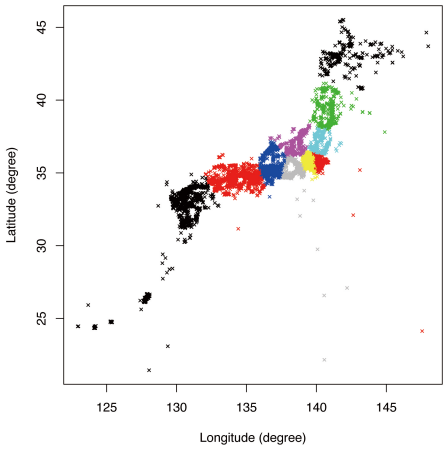

図 3.3: $K = 10$ でのクラスタリング結果

このようにして計算したクラスタは，クラスタの分割数 K が大きくなるほど，粒度が細かくなる．粒度が細かいクラスタを共有しているということは，普段使う駅，通っている学校や職場など，よりピンポイントに興味を持つ場所が似ていると考えられる．一方，粒度の粗いクラスタの共有を調べることで，東京から大阪によく出張している人など，移動する大きな地域が似ている人を判定することができる．そのため，GLoBES では，クラスタの粒度を複数用意し，それぞれに重み付けを行うことで，より正確な類似度判定を可能にした．類似度計算における重みは，クラスタの粒度が細かくなるほど大きく設定する．クラスタ分割数は任意に設定可能であるが，今回は 10，100，1000 の三段階のクラスタを設定した．表 3.3 に，クラスタの分割数とクラスタの面積の目安についてまとめた．

$K = 10$ に設定すると，関東や東北，近畿などおおよそ日本の地方区分単位のクラスタリング結果となった．$K = 100$ では，図 3.4 のように，おおよそ都道府県程度の大きさに加え，北海道や離島のような大きな面積を占める場所は，それぞれ複数に分割された結果となった．

表 3.3: クラスタ分割数と面積の目安

クラスタ分割数	クラスタの面積
10	地方区分単位
100	都道府県単位
1000	市区町村単位

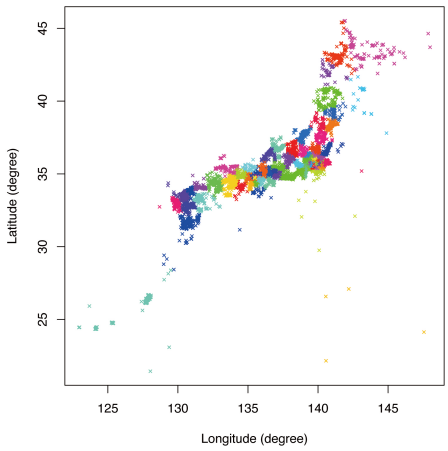

図 3.4: $K = 100$ でのクラスタリング結果

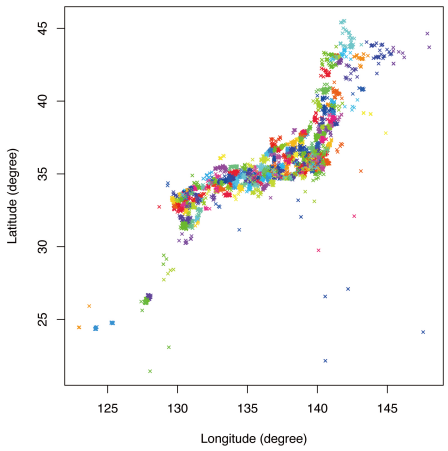

図 3.5: $K = 1000$ でのクラスタリング結果

図 3.6 では，$K = 1000$ でクラスタリングした結果を首都圏まで拡大した結果を示す．東京や神奈川の都心にかなりの発言が集中しておりクラスタが細かく分割され，千葉や群馬，栃木，茨城などの山間部においては発言がまばらで，クラスタの面積も大きくなっていることが分かる．これをさらに東京 23 区まで拡大したものが図 3.7 である．これらの結果から，クラスタリングの分割数を大きくしていくと，都心など発言の密度が高い場所ほど，面積の小さいクラスタが多く発生すると考えられる．

3.3.3 類似度判定

ユーザ類似度計算

計算したクラスタ情報を用いて，ユーザ間の類似度を計算する手法について述べる．ユーザ類似度は，すべてのユーザの組み合わせに対して設定される．例えば，ユーザ A とユーザ B が存在する場合，ユーザ A からユーザ B に対する類似度と，ユーザ B からユーザ A に対する類似度は異なるものとなる．また，ユーザ A からユーザ B に対する類似度の場合，類似度を計算する基準となるユーザ A を，基準ユーザと定義する．

クラスタは複数の粒度で計算されており，それぞれの粒度について計算した類似度に重み付けを行った加重平均を，全体の類似度とする．まずはひとつの粒度のクラスタにおける類似度計算の手順について，具体的な例を用いて説明する．

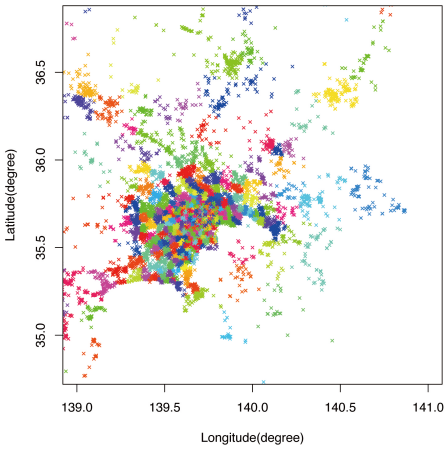

図 3.6: $K = 1000$ 首都圏

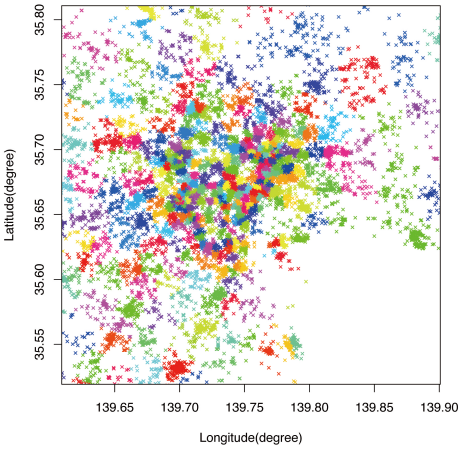

図 3.7: $K = 1000$ 東京 23 区

図 3.8 は，Taro と Hanako の 2 ユーザの位置情報付き発言にクラスタリング解析を行った例である．青丸が Taro の位置情報付き発言，赤丸が Hanako の位置情報付き発言で，近くの数字は発言 ID を表す．発言はそれぞれ K_1 から K_3 までの 3 クラスタに分類され，点線がクラスタの境界である．

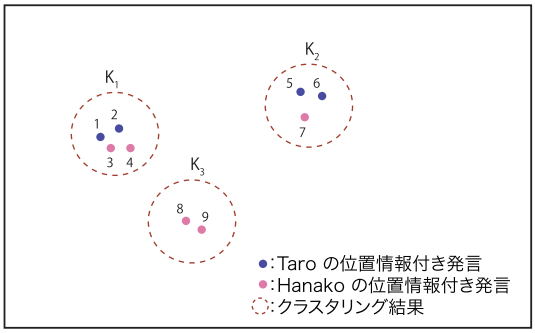

図 3.8: 位置情報付き発言のクラスタリングを行った例

Taro から Hanako に対する類似度を計算する手順について述べる．まず，ユーザがそれぞれのクラスタで何回発言しているかをカウントする．集計を行ったものを表 3.4 に示す．

表 3.4: それぞれのクラスタにおける発言数

	K_1	K_2	K_3	計
Taro	2	2	0	4
Hanako	2	1	2	5

次に，それぞれの発言数について定義する．K_1 における Taro の発言数を $t_{(K_1,\,Taro)}$ と

する．同様に，K_1 における Hanako の発言数を $t_{(K_1,\ Hanako)}$ とする．そして，K_1 において Taro と Hanako の発言数のうち少ない方を，発言共有数 $C_{K_1(Taro,\ Hanako)}$ とする．ここでは，$C_{K_1(Taro,\ Hanako)}$ は 2 となる．式 3.1 は，$C_{K_1(Taro,\ Hanako)}$ の定義を示す．

$$C_{K_1(Taro,\ Hanako)} = min(t_{(K_1,\ Taro)},\ t_{(K_1,\ Hanako)}) \tag{3.1}$$

また，すべてのクラスタにおける Taro と Hanako の発言共有数の合計を，合計発言共有数 $C_{total(Taro,\ Hanako)}$ とする．ここでは，$C_{total(Taro,\ Hanako)}$ は 3 となる．式 3.2 は，$C_{total(Taro,\ Hanako)}$ の定義であり，n はクラスタ分割数を示す．

$$C_{total(Taro,\ Hanako)} = \sum_{j=1}^{n} C_{K_j(Taro,\ Hanako)} \tag{3.2}$$

そして，Taro の全発言数を $T_{total(Taro)}$ とする．ここでは，$T_{total(Taro)}$ は 4 となる．$T_{total(Taro)}$ の定義を式 3.3 に示す．

$$T_{total(Taro)} = \sum_{j=1}^{n} t_{(K_j,\ Taro)} \tag{3.3}$$

最後に，合計発言共有数を基準ユーザである Taro の全発言数で除算した数を，Taro から Hanako に対する類似度 $S_{(Taro,\ Hanako)}$ とする．式 3.4 に，$S_{(Taro,\ Hanako)}$ の定義を示す．ここでは合計発言共有数は 3，Taro の全発言数は 4 のため，Taro から Hanako に対する類似度は 0.75 となる．

$$S_{(Taro,\ Hanako)} = \frac{C_{total(Taro,\ Hanako)}}{T_{total(Taro)}} \tag{3.4}$$

以上のアルゴリズムを一般化し，定義した式を述べる．基準ユーザ U_1 から対象となるユーザ U_2 に対する類似度を $S_{(U_1,\ U_2)}$ とする．式 3.5 は，$S_{(U_1,\ U_2)}$ を示す．

$$S_{(U_1,\ U_2)} = \frac{C_{total(U_1,\ U_2)}}{T_{total(U_1)}} \tag{3.5}$$

以上の手順で，単一粒度のクラスタにおける類似度 $S_{(U_1,\ U_2)}$ が計算される．クラスタリングは複数の分割数で行い，それぞれに重み付けをした結果を，総合類似度として計算する．図 3.9 に，複数の分割数によるクラスタリングと総合類似度の概要を示す．クラスタ分割数の小さい方から S_1，S_2 と割り当て，最大 S_m の粒度をもつ．m は最大粒度数を示す．

次に，すべての粒度の類似度に重み付けをした総合類似度 $S_{total(U_1,\ U_2)}$ を求める手順について述べる．それぞれの粒度で類似度を計算した結果を，$S_{1(U_1,\ U_2)}$, $S_{2(U_1,\ U_2)}$, ... , $S_{m(U_1,\ U_2)}$ とする．また，それぞれのクラスタ粒度に対する重みを w_1, w_2, ... , w_m とする．これらを加重平均した結果を式 3.6 に示す．

$$S_{total(U_1,\ U_2)} = \frac{w_1 \cdot S_{1(U_1,\ U_2)} + w_2 \cdot S_{2(U_1,\ U_2)} + ... + w_m \cdot S_{m(U_1,\ U_2)}}{w_1 + w_2 + ... + w_m} \tag{3.6}$$

類似ユーザ判定

前述のユーザ類似度を基に，類似ユーザを判定する方法について述べる．U_1 を基準ユーザとして考えた場合，$S_{total(U_1,\ U_2)}$ が閾値 S_{th} 以上のとき，U_1 から見た U_2 は類似ユーザであると判定する．

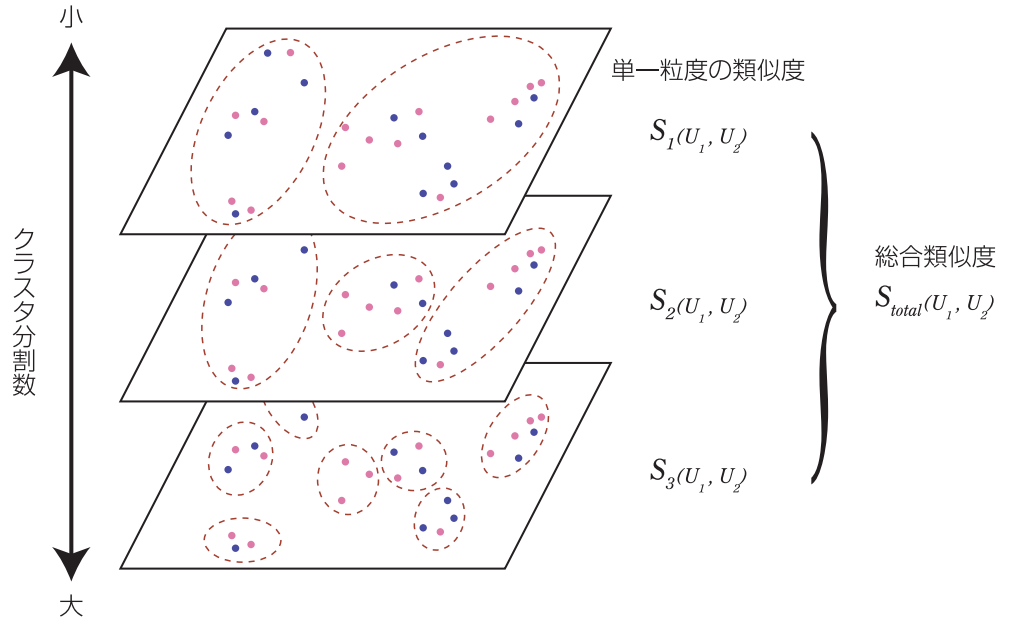

図 3.9: 複数粒度のクラスタリングと総合類似度

これを一般化すると以下のように定義される．基準ユーザを U_{from} とし，対象となるユーザを U_{to} とする．U_{from} から U_{to} に対する類似度を，$S_{total(U_{from}, U_{to})}$ と表す．このとき，$S_{total(U_{from}, U_{to})}$ が S_{th} を上回った場合，U_{from} から見た U_{to} は，U_{from} の類似ユーザであると定義する．

3.3.4 グループ判定

類似ユーザを基にしたグループ判定について述べる．U_1 を基準ユーザとして考えた場合，U_1 から見たすべての類似ユーザを，U_1 のグループであると定義する．

これを一般化すると以下のように定義される．全ユーザが n 人いる場合，U_{from} を基準ユーザとして考えると，$S_{total(U_{from}, U_1)}$，$S_{total(U_{from}, U_2)}$，… ，$S_{total(U_{from}, U_n)}$ の類似度が存在し，それぞれ S_{th} を上回ったユーザを，U_{from} から見たグループであると判定する．

3.4 モジュール構成

本節では GLoBES のモジュール構成を述べる．GLoBES は，発言取得モジュール，発言解析モジュール，クラスタリングモジュール，類似度計算モジュール，グループ情報データベース，グループ情報取得モジュールによって構成される．図 3.10 にモジュール構成図を示す．

3.4.1 発言取得モジュール

発言取得モジュールの詳細について述べる．発言取得モジュールは，ソーシャルメディアの公開している API から位置情報付きの発言を取得し，発言解析モジュールへと渡す．モ

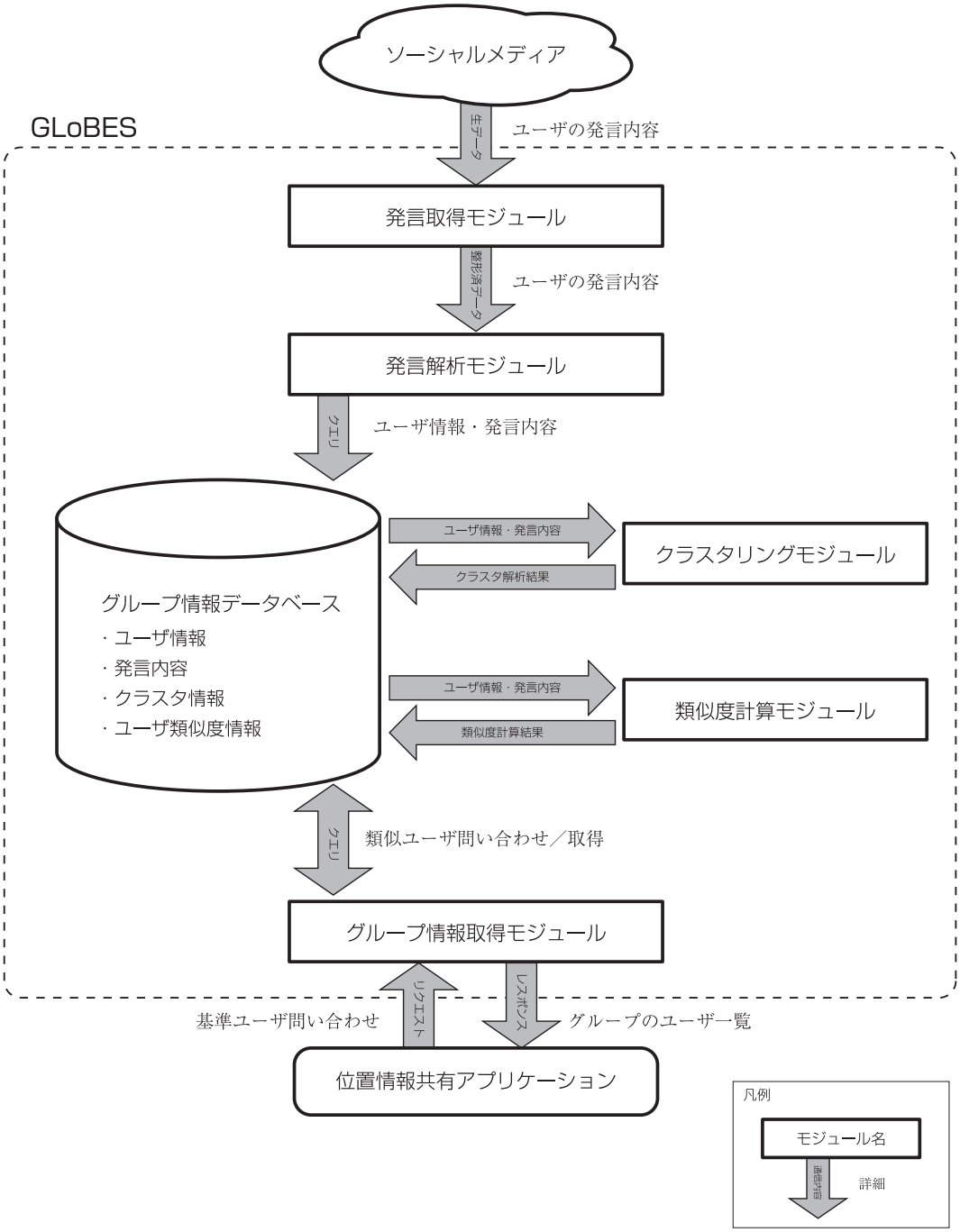

図 3.10: モジュール構成図

ジュール内は，ソーシャルメディア認証部，発言フィルタリング部，データ形式変換部に分割されている．図 3.11 を踏まえ，それぞれについて動作概要を述べる．

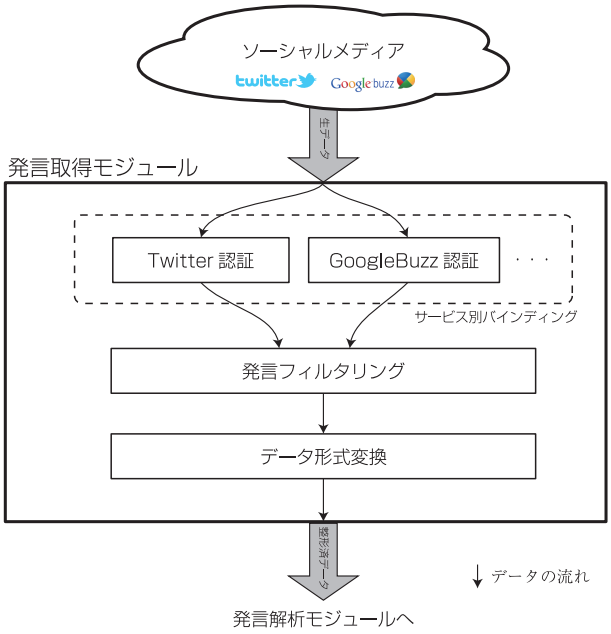

図 3.11: 発言取得モジュール

ソーシャルメディア認証部

ソーシャルメディア認証部では，様々なソーシャルメディアサービスにおける API の差異を吸収するバインディングの役割を果たす．対象となるソーシャルメディアは，発言を取得する API が公開されていることと，発言に位置情報を付加できることが必要条件である．表 3.5 では既存のソーシャルメディアを比較した．GLoBES が対応可能なサービスは Twitter や GoogleBuzz などであるが，必要条件を満たすサービスであれば．サービス別バインディングを追加することで対応可能である．

表 3.5: 対象とするソーシャルメディアの比較

サービス名	情報取得 API	位置情報の付加	対応の可否
Twitter[29]	公開 (REST/Streaming)	○	○
GoogleBuzz[9]	公開 (REST/Streaming)	○	○
mixi ボイス	非公開（投稿のみ公開）	×	×
mixi チェックイン	非公開（投稿のみ公開）	○	×

発言フィルタリング部

発言フィルタリング部では，ソーシャルメディアから取得した発言の中から，解析に不要な発言を取り除く処理を行う．ソーシャルメディアの API から取得した情報には，位置情報の付加されていない発言や，エラーのある発言などが含まれている場合がある．解析のためには，このような発言を取り除く必要がある．

データ形式変換部

データ形式変換部では，フィルタリングされたデータを解析可能な形式に変換する．対象とするソーシャルメディアにより，取得できる情報の種類や形式は異なる．それらの中から，ユーザ情報と発言内容データのみを抽出し，発言解析モジュールへと送信する．

3.4.2 発言解析モジュール

発言解析モジュールの詳細について述べる．発言解析モジュールは，整形されたソーシャルメディアの発言情報を発言取得モジュールから受け取り，ユーザの情報と発言内容の情報とに分離し，グループ情報データベースへ保存する．モジュール内は，ユーザデータ／発言内容分離部，クエリ生成部に分割されている．図 3.12 を踏まえ，詳細について述べる．

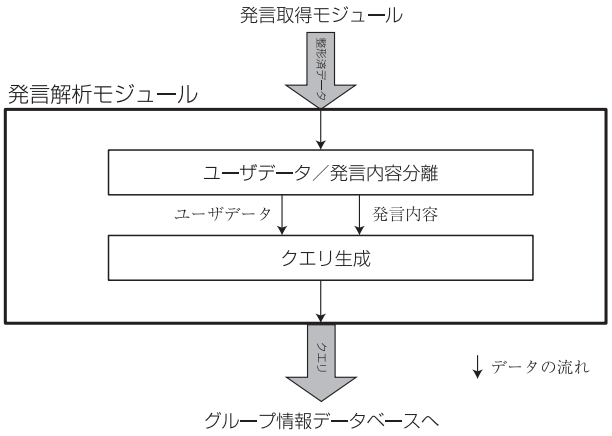

図 3.12: 発言解析モジュール

ユーザデータ／発言内容分離部

ユーザデータ／発言内容分離部では，発言取得モジュールから送られたデータを，データベースへ格納するのに適したデータ構造に変換する．発言取得モジュールからは，発言内容と発言したユーザの情報が一体となったデータを受け取る．一方，グループ情報データベースでは，効率的な解析のために発言内容とユーザの情報は別のテーブルに分割されている．そのテーブル構造に適したように，発言内容とユーザ情報を分離し，クエリ生成部へと送信する．

クエリ生成部

クエリ生成部では，ユーザデータ／発言内容取得部から送られたデータをグループ情報データベースに格納する．

3.4.3　クラスタリングモジュール

クラスタリングモジュールの詳細について述べる．クラスタリングモジュールは，グループ情報データベースに保存されている発言内容を取得し，位置情報を基にクラスタリング解析を行い，結果を再びグループ情報データベースに保存する．モジュール内は，発言内容取得部，クラスタリング解析部，クエリ生成部に分割されている．

図 3.13 を踏まえ，詳細について述べる．

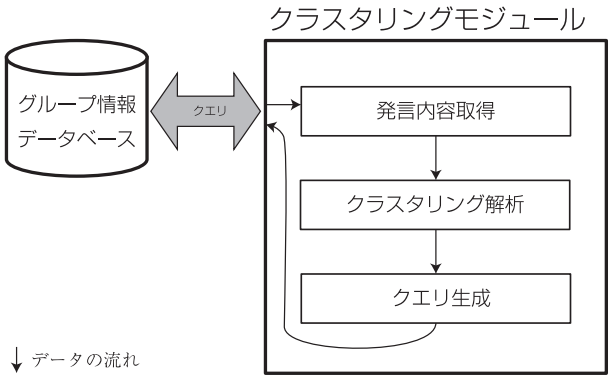

図 3.13: クラスタリングモジュール

発言内容取得部

発言内容取得部では，グループ情報データベースから発言内容データを取得する．クラスタリングモジュール実行時，クラスタリング解析を行う期間の日付を受け取る．

クラスタリング解析部

クラスタリング解析部では，発言内容取得部から受け取った発言データを，K-means 法を用いてクラスタリング解析を行い，結果をクエリ生成部へ送信する．K-means 法ではクラスタ分割数を指定する必要があるため，クラスタリングモジュール実行時に設定する．

クエリ生成部

クエリ生成部では，クラスタリング解析部から受け取ったデータを，グループ情報データベースへ格納する．

3.4.4　類似度計算モジュール

類似度計算モジュールの詳細について述べる．類似度計算モジュールは，グループ情報データベースに保存されているユーザデータと発言内容を取得し，類似度計算を行い，結果を再びグループ情報データベースに保存する．モジュール内は，ユーザデータ／発言内容取得部，類似度計算部，クエリ生成部に分割されている．

図 3.14 を踏まえ，詳細について述べる．

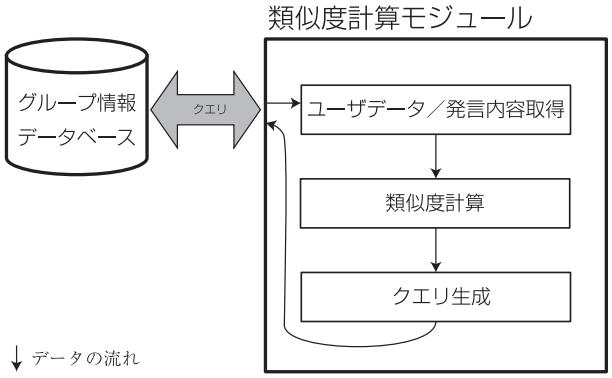

図 3.14: 類似度計算モジュール

ユーザデータ／発言内容取得部

ユーザデータ／発言内容取得部では，グループ情報データベースから発言内容データを取得し，結果を類似度計算部に送信する．類似度計算モジュール実行時，類似度計算を行う対象のユーザ ID リストを受け取る．

類似度計算部

類似度計算部では，ユーザデータ／発言内容取得部から送られたデータを用い，類似度計算を行う．対象となるすべてのユーザの組み合わせに対して類似度計算を行い，結果をクエリ送信部へと送る．

クエリ生成部

クエリ生成部では，類似度計算部から受け取ったデータをグループ情報データベースに格納する．

3.4.5 グループ情報取得モジュール

グループ情報取得モジュールの詳細について述べる．グループ情報取得モジュールは，位置情報共有アプリケーションの要求に応じて，グループ情報データベースからグループのユーザ一覧を取得し，位置情報共有アプリケーションに応答する．モジュール内は，リクエスト解析部，クエリ生成部，結果取得部，ユーザリスト生成部，レスポンス生成部に分割されている．図 3.15 を踏まえ，詳細について述べる．

リクエスト解析部

リクエスト解析部では，アプリケーションからリクエストを受け取り，解析する．基準ユーザの ID を，クエリ生成部へ送信する．

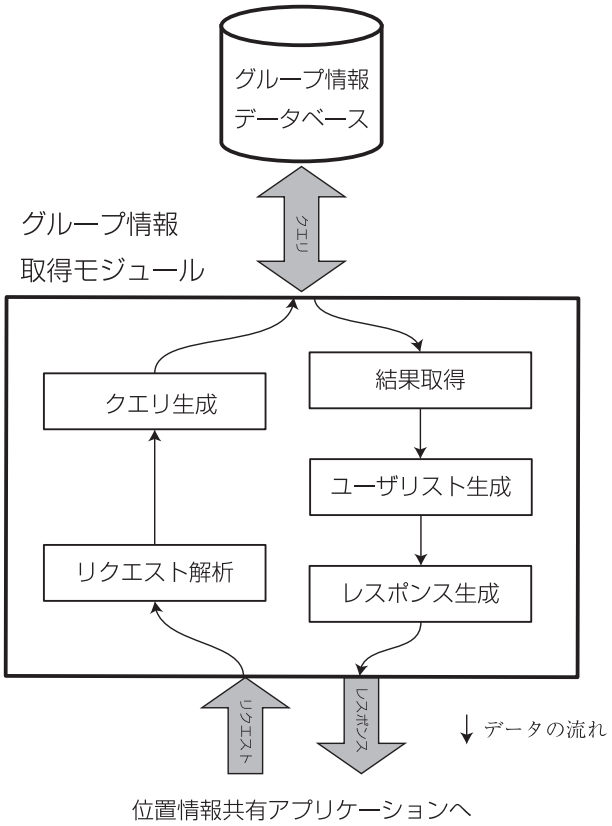

図 3.15: グループ情報取得モジュール

クエリ生成部

クエリ生成部では，リクエスト解析部から基準ユーザ ID を受け取り，グループ情報データベースにクエリを送信する．

結果取得部

結果取得部では，グループ情報データベースから類似度情報を取得し，ユーザリスト生成部へ送信する．

ユーザリスト生成部

ユーザリスト生成部では，結果取得部から受け取ったデータの中から類似度が閾値以上のユーザのみを抽出し，レスポンス生成部へ送信する．結果取得部からはユーザ ID と類似度を受け取り，類似度が S_{th} より大きいユーザを抽出する．

レスポンス生成部

レスポンス生成部では，結果取得部から受け取ったユーザ ID のリストを，アプリケーションへ送信する．

3.4.6　グループ情報データベース

グループ情報データベースの詳細について述べる．グループ情報データベースは，GLoBES におけるすべての情報を保存するリレーショナルデータベースである．ソーシャルメディアから取得した発言情報や各モジュールによる計算結果を格納する．

3.5　本章のまとめ

本章では，本研究の提案する GLoBES の設計について述べた．

まず，GLoBES の概要について述べ，本研究の想定環境についてまとめた．グループ判定アルゴリズムの節では，ソーシャルメディアから位置情報付きの発言を取得し，位置情報を基に発言のクラスタリングを行い，類似度を判定するという流れと，それぞれの詳細について述べた．モジュール構成の節では，GLoBES が発言取得モジュール，発言解析モジュール，クラスタリングモジュール，類似度計算モジュール，グループ情報データベース，グループ情報取得モジュールに分割されていることと，それぞれの動作概要について述べた．

第4章　実装

本章では，GLoBES の実装について述べる．まず概要を述べ，実装環境をまとめる．次に，グループ判定アルゴリズムの実装と，各モジュールの実装について述べる．

4.1 概要

本節では GLoBES 実装における概要について述べる．まず，GLoBES が対象とするソーシャルメディアの要件として，以下の二点を挙げる．

1. 発言内容が API により取得可能なこと
2. 発言に位置情報が付加されうること

1 番目は，ソーシャルメディアの発言を取得する必要があるため，API が公開されているソーシャルメディアを対象とする．2 番目の要件は，発言の位置情報を基にクラスタリング解析を行うため，それぞれの発言に緯度経度の位置情報が付加されていることを必要とする．

以上の要件を考慮し，今回対象としたソーシャルメディアについて述べる．発言を取得する API が公開されており，発言に位置情報を付加できるソーシャルメディアとしては，具体的には Twitter や GoogleBuzz が候補となった．今回は発言取得 API の充実や，サービス利用者の多さから Twitter を対象として実装を行った．

4.2 実装環境

本節では GLoBES の実装環境について述べる．

4.2.1 ハードウェア構成

ハードウェアの要件と構成を述べる．GLoBES はソーシャルメディアの発言を取得するため，常時ネットワークに接続されている必要がある．また，類似度計算に多くのメモリと CPU パワーを必要とするため，これらの性能が可能な限り高いハードウェアが望ましい．

4.2.2 ソフトウェア構成

ソフトウェアの選定と構成について述べる．実装に使用したソフトウェアを表 4.1 にまとめた．図 4.1 では，各ソフトウェアの連携について示す．

グループ情報取得モジュールでは Web ベースの API を提供するため，Web サーバとして実績のある Apache HTTP Server 2.2.9 を選定した．また，グループ情報データベースには RDBMS である MySQL 5.0.51a を選定した．各モジュールの実装には，スクリプト言語である PHP 5.2.6 を選定し，クラスタリングモジュールにおいては R 言語 ver. 2.7.1 を使用した．

GLoBES は常に発言を取得するため，長期間の安定稼働が必要とされる．そのため，これらのソフトウェアを実行させる OS として Debian GNU/Linux 5.0.7 を使用した．

4.3 グループ判定アルゴリズムの実装

本節ではグループ判定アルゴリズムの実装について述べる．

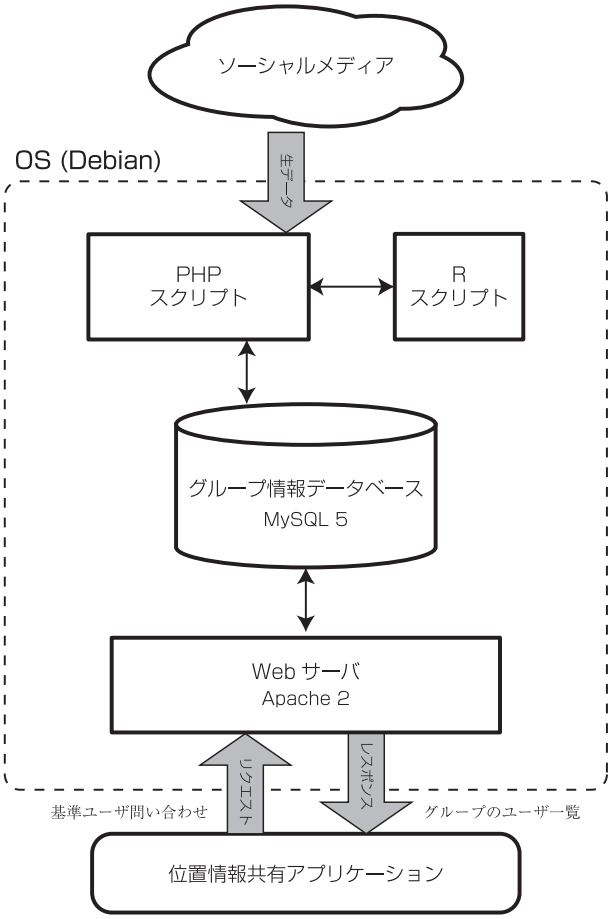

図 4.1: 各ソフトウェアの連携

表 4.1: ソフトウェア構成

要素	詳細
OS	Debian GNU/Linux 5.0.7(lenny)
Web サーバ	Apache HTTP Server 2.2.9
解析スクリプト	PHP ver. 5.2.6 with Suhosin-Patch 0.9.6.2
データベース	MySQL ver. 5.0.51a
クラスタリング解析	R ver. 2.7.1

4.3.1 ソーシャルメディアにおける位置情報付き発言の取得

今回は Twitter における位置情報付きの発言を取得した．Twitter の StreamingAPI から発言を取得し，グループ情報データベースに保存する．発言の取得は，発言取得モジュール，発言解析モジュールが行う．実装についての詳細は 4.4.1 と 4.4.2 において述べる．

4.3.2 発言のクラスタリング

取得した発言を，K-means 方を用いてクラスタリング解析する．解析は，指定した期間に取得した発言について行う．

解析の実行には，R 言語を用いた．詳細は 4.4.3 において述べる．

4.3.3 類似度判定

類似度判定の実装について述べる．類似度判定は，発言数カウント，ユーザ類似度計算の 2 段階に分かれている．複数のクラスタ分割数でクラスタリング解析を行った場合は，それぞれについて類似度判定を行う．

まず発言数カウントのアルゴリズム *CountTweets* について，Algorithm 1 に示す．クラスタリング解析された発言は，それぞれ *userId*, *clusterId* を含み，それらの集合を T とする．$T_i.userId$, $T_i.clusterId$ は，i 番目の発言における *userId*, *clusterId* を表す．解析対象となるユーザ数を *usersNum* とし，クラスタ分割数を *clusterNum* とする．これらの T, *usersNum*, *clustersNum* を引数として与え，ユーザがどのクラスタで何回発言しているかという二次元配列 *counts*[*userId*][*clusterId*] を取得する．

Algorithm 1 CountTweets(T, *usersNum*, *clustersNum*)

Input: クラスタリング済の発言 T, 全ユーザ数 *usersNum*, 全クラスタ数 *clustersNum*

Output: 各クラスタにおけるユーザの発言数 *counts*[*userId*][*clusterId*]

```
1: i = 0, tweetsNum = |T|;
2: counts[usersNum][clustersNum];
3: while i < tweetsNum do
4:     counts[T_i.userId][T_i.clusterId] := COUNT[T_i.userId][T_i.clusterId] + 1;
5:     i := i + 1;
6: end while
7: return counts[userId][clusterId]
```

次に，ユーザ類似度計算のアルゴリズム *CalculateSimilarity* について，Algorithm 2 に示す．*CountTweets* から出力された *counts*[*userId*][*clusterId*] と，解析対象となるユーザ数 *usersNum* を引数として与える．基準ユーザ $fromUserId$ から対象ユーザ $toUserId$ に対する類似度を S とし，$S.fromUserId$, $S.toUserId$, $S.similarity$ の属性を持つ．

また，$SL.insert(S)$ の操作によって集合 SL に追加する．対象となるすべてのユーザの組み合わせに対して類似度を計算した結果 SL を取得する．類似度計算において，$fromTotal$ は基準ユーザの全発言数，*shared* はあるクラスタにおける基準ユーザと対象ユーザの発言共有数を示す．

Algorithm 2 CalculateSimilarity($counts[userId][clusterId]$, $usersNum$)

Input: 各クラスタにおけるユーザの発言数 $counts[userId][clusterId]$, 全ユーザ数 $usersNum$

Output: 各ユーザ間における類似度の集合 $SL = \{S\}$

```
1:  i = 0, j = 0;
2:  while i < usersNum do
3:      while j < usersNum do
4:          if i = j then
5:              break;
6:          end if
7:          fromTotal = 0, shared = 0;
8:          while c < clustersNum do
9:              fromTotal := fromTotal + counts[i][c];
10:             shared := shared + min(counts[i][c], counts[j][c]);
11:         end while
12:         S.fromUserId := i;
13:         S.toUserId := j;
14:         S.similarity := shared/fromTotal;
15:         SL.insert(S);
16:         j := j + 1;
17:     end while
18:     i := i + 1;
19: end while
20: return SL
```

4.3.4 グループ判定

グループ判定部の実装について述べる．グループ判定のアルゴリズム *DetectGroup* について，Algorithm 3 に示す．

今回は，クラスタ分割数を 10，100，1000 に設定してクラスタリング解析を行い，類似度判定を行ったデータについてグループ判定を行うことを想定する．SL_{10} はクラスタ分割数 10 のデータで *CalculateSimilarity* から出力された各ユーザ間における類似度の集合である．同様に SL_{100}，SL_{1000} はクラスタ分割数 100，1000 における類似度の集合である．また，クラスタ分割数 10，100，1000 に対する重み付けの値を，$weight_{10}$，$weight_{100}$，$weight_{1000}$ とする．さらに，類似ユーザであると見なす類似度の閾値を *similarThresh* とする．*usersNum* は解析対象となるユーザ数を示す．これらの値を引数として与える．

グループ判定結果を G とし，基準ユーザの ID である *G.userId*，グループであるユーザ ID の集合 *G.groupIds* の属性を持つ．*G.groupIds.insert*(*userId*) の操作によって，グループと判定したユーザ ID を *G.groupIds* に追加する．また，*GL.insert*(*G*) の操作によって，G をグループ判定結果の集合である GL に追加する．

Algorithm 3 DetectGroup(SL_{10}, SL_{100}, SL_{100}, *usersNum*, *similarThresh*, $weight_{10}$, $weight_{100}$, $weight_{1000}$)

Input: SL_{10}，SL_{100}，SL_{1000}，全ユーザ数 *usersNum*，グループ判定閾値 *similarThresh*, $weight_{10}$, $weight_{100}$, $weight_{1000}$

Output: グループ判定結果 $GL = \{G\}$

```
1:  i = 0, j = 0;
2:  while i < usersNum do
3:      G.userId := j;
4:      while j < usersNum do
5:          if i = j then
6:              break;
7:          end if
8:          {SL から fromUserId と toUserId をキーにして S を検索する}
9:          S_10 := (SL_10.fromUserId = i && SL_10.toUserId = j);
10:         S_100 := (SL_100.fromUserId = i && SL_100.toUserId = j);
11:         S_1000 := (SL_1000.fromUserId = i && SL_1000.toUserId = j);
12:         totalSimilarity := (S_10 * weight_10 + S_100 * weight_100 + S_1000 * weight_1000) / (weight_10 +
            weight_100 + weight_1000);
13:         if totalSimilarity > similarThresh then
14:             G.groupIds.insert(j);
15:         end if
16:         j := j + 1;
17:     end while
18:     GL.insert(G);
19:     i := i + 1;
20: end while
21: return GL
```

4.4 各モジュールの実装

本節では各モジュールの実装について述べる．

4.4.1 発言取得モジュール

発言取得モジュールの実装について述べる．発言取得モジュールは，ソーシャルメディア認証部，発言フィルタリング部，データ形式変換部の三つに分割されている．実装言語は PHP を利用した．

ソーシャルメディア認証部

今回，Twitter の発言を取得するために，StreamingAPI(statuses/filter)[28] を利用した．以下のパラメータで StreamingAPI に接続し，日本周辺の位置情報付き発言をリアルタイムで取得する．URL は Twitter におけるパブリックな発言を取得する API で，日本周辺を 8 つの矩形に分割し，緯度経度を指定することで，日本周辺の位置情報付き発言のみを取得している．Twitter のアカウント名とパスワードで Basic 認証を行い，パラメータである Content は POST メソッドで送信する．

- URL: $http://stream.twitter.com/1/statuses/filter.json$
- Method: $POST$
- Content: $'locations' => '139, 20.42, 148.75, 45.55, 132, 20.42, 139, 39, 129, 20.42, 132, 35, 122.93, 20.42, 129, 33'$

発言フィルタリング部

Twitter の StreamingAPI では，発言された情報だけではなく，削除された発言の情報や，重複した情報が取得される場合がある．このような発言をエラーとして除去し，データ形式変換部へと送信する．

データ形式変換部

データ形式変換部では，フィルタリングされた発言を JSON 形式に変換する．

4.4.2 発言解析モジュール

発言解析モジュールの実装について述べる．実装言語は PHP を利用した．

ユーザデータ／発言内容分離部

ユーザデータ／発言内容分離部では JSON データをパースし，ユーザ情報テーブルと発言内容テーブルに必要なデータを分割して，クエリ生成部へと送信する．

ユーザデータは，id，id_str，name の 3 項目に分割する．id はグループ情報データベースにおいてユーザを識別する ID で，新しいユーザの発言を取得するたびに付与する．id_str は，Twitter 内部でユーザを識別するための一意な ID で，name は Twitter の screen_name である．

表 4.2: ユーザデータ

項目	詳細
id	DB におけるユーザ ID
id_str	Twitter におけるユーザ ID
name	Twitter の screen_name

発言内容は，id，user_id，tweet_str，name の 4 項目に分割する．id はグループ情報データベースにおいて発言を識別する ID である．user_id はユーザデータの id を指し，グループ情報データベース内で発言とユーザデータを連携させるために用いる．tweet_id は Twitter 内部で発言を識別するための一意な ID である．latitude，longitude は発言に付与された緯度経度の座標であり，text は発言の本文である．orig_created_at は，Twitter に発言が投稿された日時を示す．

表 4.3: 発言内容

項目	詳細
id	DB における発言 ID
user_id	DB におけるユーザ ID
tweet_id	Twitter における発言 ID
latitude	発言に付与された緯度座標
longitude	発言に付与された経度座標
text	発言内容
orig_created_at	発言された日時

クエリ生成部

クエリ生成部では，ユーザデータ／発言内容分離部から受け取ったデータをクエリに変換する．グループ情報データベースと通信するために，PHP の O/R マッパーである Doctrine 1.2.3 を用いた．ソースコード 4.1 は，クエリ生成部がユーザデータ／発言内容をデータベースに格納する動作を示す．

ソースコード 4.1: クエリ生成部

```
$json = # 発言取得モジュールから受信
$t = json_decode($json, true);

$user = new User();
$user->setName($t['user']['screen_name']);
$user->setIdStr($t['user']['id_str']);
$user->save();

$tweet = new Tweet();
$tweet->setUserId($user->getId());
$tweet->setTweetId($t['id_str']);
$tweet->setLatitude($t['geo']["coordinates"][0]);
$tweet->setLongitude($t['geo']["coordinates"][1]);
$tweet->setText($t['text']);
$tweet->setOrigCreatedAt(date('Y-m-d H:i:s', strtotime($t['created_at'])));
$tweet->save();
```

4.4.3 クラスタリングモジュール

クラスタリングモジュールの実装について述べる．発言内容取得部，クエリ生成部の実装言語は PHP を利用した．

発言内容取得部

まず必要とする期間の発言内容を，グループ情報データベースから取得する．MySQL の SELECT 文を実行し，結果を TSV 形式で取得する．発言内容は標準出力に出力し，テキストデータとして一時的に保存する．

クラスタリング解析部

クラスタリング解析部では，R 言語を用いて解析を行う．クラスタリング解析に使用する発言内容は，id，user_id，tweet_id，name，latitude，longitude，text，orig_created_at，created_at，updated_at の 9 項目からなるリストである．具体的なエントリの例を，ソースコード 4.2 に示す．

ソースコード 4.2: tweet.tsv

```
650916  385  209814109615636 48  hiru_ecn  35.35729400  139.63366300  2011-01-01 08:15:32
2011-01-01 08:14:47  2011-01-01 08:14:47
```

R 言語を用いて，発言のクラスタリングを行う．ここでは，クラスタ分析数を 10，100，1000 の 3 パターンで行い，結果を出力する R のプログラムをソースコード 4.3 に示す．解析に必要な緯度経度情報のみを取り出した行列を作成し，R の kmeans() 関数でクラスタリングを行い，整形した結果を保存している．

ソースコード 4.3: クラスタリング解析部

```
orig <- matrix(scan("./tweet.tsv", sep="\t", what=character(), quote=""), ncol=10, byrow=T)
data <- matrix(as.numeric(orig[,5:6]), ncol=2)

k <- 10
km <- kmeans(data, k)
result <- matrix(c(rep("\\n", nrow(orig)), orig[,1], orig[,2], rep(k, nrow(orig)), km$
cluster, rep("k-means", nrow(orig)), km$centers[,1][km$cluster], km$centers[,2][km$cluster
], rep("\\n", nrow(orig)), rep("\\n", nrow(orig))), ncol=10)
write(t(result), file="./cluster.txt", ncolumns=10, sep="\t")
```

```
8
9  k <- 100
10 km <- kmeans(data, k, iter.max = 100)
11 result <- matrix(c(rep("\\n", nrow(orig)), orig[,1], orig[,2], rep(k, nrow(orig)), km$
   cluster, rep("k-means", nrow(orig)), km$centers[,1][km$cluster], km$centers[,2][km$cluster
   ], rep("\\n", nrow(orig)), rep("\\n", nrow(orig))), ncol=10)
12 write(t(result), file="./cluster.txt", ncolumns=10, sep="\t", append=T)
13
14 k <- 1000
15 km <- kmeans(data, k, iter.max=100)
16 result <- matrix(c(rep("\\n", nrow(orig)), orig[,1], orig[,2], rep(k, nrow(orig)), km$
   cluster, rep("k-means", nrow(orig)), km$centers[,1][km$cluster], km$centers[,2][km$cluster
   ], rep("\\n", nrow(orig)), rep("\\n", nrow(orig))), ncol=10)
17 write(t(result), file="./cluster.txt", ncolumns=10, sep="\t", append=T)
```

クラスタリング結果は，id，tweet_id，user_id，level，cluster_no，method，center_lat，center_lng，created_at，updated_at の 10 項目からなるリストである．表 4.4 に，それぞれの値の詳細を示す．ソースコード 4.4 は，具体的なエントリの例である．

表 4.4: クラスタリング結果

項目	詳細
id	DB における発言 ID
tweet_id	Twitter における発言 ID
user_id	DB におけるユーザ ID
level	クラスタ分割数
cluster_no	分類されたクラスタ ID
method	クラスタリングに用いたアルゴリズム
center_lat	クラスタ中心の緯度
center_lng	クラスタ中心の経度
created_at	データベースに保存した日時
created_at	情報を更新した日時

ソースコード 4.4: cluster.tsv

```
1 \n  650916  385  10  6  k-means  35.6575151159336  139.651827804471  \n  \n
```

クエリ生成部

クエリ生成部では，クラスタリング解析結果を受け取り，グループ情報データベースのクラスタリング結果テーブルに INSERT 文を発行する．

4.4.4 類似度計算モジュール

類似度計算モジュールの実装について述べる．モジュール全体の実装言語は PHP を利用した．

ユーザデータ／発言内容取得部

ユーザデータ／発言内容取得部では mysql コマンドを用いてグループ情報データベースと接続する．必要とするユーザ情報，発言内容を取得するため，グループ情報データベースに SELECT 文を発行し，結果を類似度計算部に送信する．

類似度計算部

類似度計算部では，ユーザデータ／発言内容取得部から受け取ったデータを解析し，結果をクエリ生成部へと送信する．類似度計算結果は，基準ユーザ ID，対象ユーザ ID，類似度の 3 項目からなるリストである．

クエリ生成部

クエリ生成部では，グループ情報データベースの類似度テーブルに INSERT 文を発行する．

4.4.5 グループ情報取得モジュール

グループ情報データベースの実装について述べる．モジュール全体の実装言語は PHP を利用した．

リクエスト解析部

リクエスト解析部においては，Web サーバの実装である Apache HTTP Server 2.2.9 を利用した．その上で，PHP5 用のフレームワークである Symfony 1.4.8 を利用して WebAPI を実装した．リクエストは HTTP の GET メソッドを用い，パラメータに基準ユーザ情報を受け取る．

アプリケーションからグループ情報を利用するため，以下のような API を作成した．

URL

http://aoi.ht.sfc.keio.ac.jp/geo/index.php/location/group/

Method

GET

Parameter

user_name

引数の user_name には，グループを取得したいユーザの Twitter における screen_name を指定する．例えば，hiru_ecn のグループを取得したい場合，以下の URL にアクセスする．

`http://aoi.ht.sfc.keio.ac.jp/geo/index.php/location/group/screen_name/hiru_ecn`

クエリ生成部

クエリ生成部では Doctrine を用いてグループ情報データベースと接続する．グループ情報データベースの類似度情報テーブルに，基準ユーザ ID をキーとする SELECT 文を発行する．

結果取得部

結果取得部では，Doctrine を用いてグループ情報データベースと接続する．クエリの実行結果を受け取り，ユーザリスト生成部へ送信する．

ユーザリスト生成部

ユーザリスト生成部では，結果取得部から受け取ったデータの中から類似度が一定以上のユーザのみを抽出し，レスポンス生成部へ送信する．結果はグループとなるユーザ ID のリストである．

レスポンス生成部

レスポンスは HTTP リクエストに対するレスポンスとして送信する．結果に応じて適切な HTTP ステータスコードを送信し，結果をメッセージボディに格納する．リクエストの形式が不正であった場合は 400 Bad Request を送信し，リクエストを受けた基準ユーザが存在しなかった場合は 404 Not Found を送信する．ユーザリストが正しく取得できた場合は，200 OK と結果を送信する．ユーザリストは JSON 形式にエンコードして送信する．

4.4.6 グループ情報データベース

グループ情報データベースの実装について述べる．リレーショナルデータベースには MySQL 5.0.51a を使用した．

発言解析モジュール，クラスタリングモジュール，類似度計算モジュール，グループ情報取得モジュールと SQL を用いて通信する．発言内容，ユーザ情報，クラスタリング結果，類似度の 4 つのテーブルに分割される．

4.4.7　位置情報共有アプリケーション例

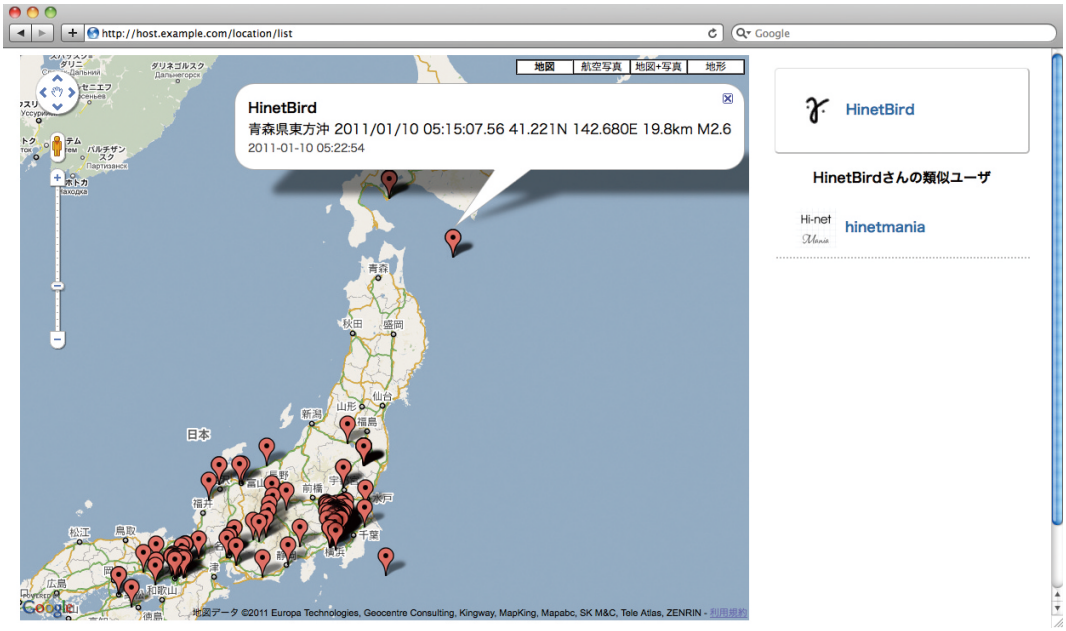

図 4.2: 位置情報共有アプリケーション例

実際の位置情報共有アプリケーションの例として，グループ情報を用いて類似するユーザを提示するアプリケーションを実装した．図 4.2 に，アプリケーション例のスクリーンショットを示す．

左のマップには取得した位置情報付き発言のうち最新の 200 件がプロットされており，マーカをクリックするとユーザ名，発言内容，発言時刻の詳細表示がポップアップされる．また，右のメニューにはマップで詳細表示されているユーザが上に表示され，そのユーザのグループであるユーザ一覧がその下にリスト表示されている．ユーザ名をクリックすると，そのユーザを基準ユーザとしたグループが表示される．

4.5　本章のまとめ

本章では，GLoBES の実装について述べた．概要と実装環境をまとめ，グループ判定アルゴリズムの実装と，各モジュールの実装について述べた．

第5章　評価

本章では，GLoBES の評価を行う．クラスタリングモジュール，類似度計算モジュールについてそれぞれ評価方針を述べ，評価結果を示す．

5.1 クラスタリングモジュールの評価

本節では，クラスタリングモジュールの評価を行う．評価方針としては，クラスタリング対象となる発言数とクラスタリング解析処理に必要な時間との関係を用い，評価結果を述べる．

5.1.1 評価方針

クラスタリングモジュールは，クラスタリング対象となる発言数と，クラスタリング解析処理に必要な時間との関係を評価する．事前実験の結果，1 日の位置情報付き発言は日本周辺に限定するとおよそ 20000 件取得されることが分かった．クラスタリングモジュールは数日〜数週間に一度実行され，取得した一定期間の発言に対して解析を行う．そのため，解析対象となる発言数が増加しても計算可能であることが求められる．

5.1.2 評価結果

2000 件，20000 件，200000 件の発言数に対し，それぞれ 10 回クラスタリング計算を試行し，実行にかかった時間と実行時間の標準偏差を表 5.1 にまとめた．クラスタリング計算では，すべての発言を読み込み，$K = 10$，$K = 100$，$K = 1000$ の 3 つの粒度のクラスタリング結果を出力している．

表 5.1: 発言数とクラスタリング解析時間における評価

発言数	平均解析時間（秒）	標準偏差（秒）
2000	0.77	0.02
20000	5.05	0.29
200000	50.51	3.36

表 5.1 の結果から，約 1 日分の発言数である 20000 件のデータが，平均 5.05 秒で解析可能であることが分かる．また，計算量は $O(n)$ であり，問題無く定期的に解析を行うことができる．

5.2 類似度計算モジュールの評価

本節では類似度計算モジュールの評価について述べる．評価方針としては，解析対象となる発言数と類似度解析に必要な時間との関係を用い，評価結果を述べる．

5.2.1 評価方針

類似度計算モジュールは，解析対象となる発言数と，解析に必要な時間との関係を評価する．事前にクラスタリング解析を行ったデータを用い，類似度計算に要した時間を測定する．

5.2.2 評価結果

一定の期間の発言を抽出し，クラスタリング解析を行った後のデータに対し，類似度計算を実行した．対象とした発言の期間は，2011 年 1 月 1 日 0:00 から 30 分，60 分，120 分間で，それぞれ 1677 件，2740 件，4389 件の発言を抽出した．対象とする発言数と，期間内におけるユニークなユーザ数，解析時間を表 5.2 に示す．

表 5.2: 発言数と類似度計算時間における評価

解析期間（分）	発言数	ユニークユーザ数	解析時間（秒）
30	1677	1107	409.247
60	2740	1603	859.287
120	4389	2211	1669.050

表 5.2 の結果から，計算量は $O(n)$ であり，解析対象とする発言の期間内に類似度計算を完了させることが可能であると言える．

5.3 本章のまとめ

本章では，GLoBES の評価を行った．クラスタリングモジュールの評価では，約一日分の位置情報付き発言である 20000 件の発言を 5.05 秒で解析可能なことを示し，解析対象となる発言数が増加しても計算可能であることを述べた．また，類似度計算モジュールの評価では，解析対象となる時間が増加しても，解析対象期間以下の時間で類似度計算が完了可能なことを述べた．

第6章　関連研究

本章では，本研究と関連研究との比較を行う．位置情報処理における研究では，類似したユーザの判定について既存の手法を挙げ，本研究の利点について述べる．また，位置情報共有アプリケーションにおいて重要なプライバシ問題について述べ，関連研究における手法を比較する．

6.1 位置情報処理に関する研究

本節では，位置情報共有アプリケーションにおける課題について，類似したユーザの判定，プライバシの観点から言及する．

6.1.1 類似したユーザの判定

社会性重視のアプリケーションにおいて，類似ユーザを検出する機能は重要である．類似ユーザ情報を用いることで，新しい友達候補を提示したり，広告を配信したりするサービスが可能になる．類似ユーザ検出において，既存の手法と GLoBES を表 6.1 において比較した．比較項目は，ユーザの位置情報を利用しているか，リアルタイム性があるか，プライバシに対する問題があるかの 3 項目である．リアルタイム性とは，日々変化するユーザ間のつながりや行動パターンに対して，類似ユーザ検出結果がリアルタイムに反映されるかどうかを指す．

Twitter[29] や Facebook[6] などでは，既存のユーザ同士のつながりや，ユーザのプロフィールなどを解析することにより，類似ユーザの提示を行っている．これらを，既存のソーシャルグラフを用いた手法とする．また，GPS ロガー端末をユーザに持たせ，GPS の軌跡から類似したユーザを判定する研究も行われている [16]．これを，GPS ロガーによる軌跡の解析を用いた手法とする．

表 6.1: 既存の類似ユーザ検出手法と GLoBES の比較

手法	ユーザの位置情報を利用	リアルタイム性	プライバシに対する問題
既存のソーシャルグラフを用いた手法	×	○	少ない
GPS ロガーによる軌跡の解析を用いた手法	○	×	多い
携帯電話の GPS ログの解析を用いた手法	○	△	多い
GLoBES	○	あり	少ない

第一に，既存のソーシャルネットワークにおける類似ユーザの検出では，オンライン上のユーザ間の関係のみしか参照していないため，実世界におけるユーザの行動は反映されていない．さらに，Twitter[29] などの一方通行にフォローが可能なソーシャルネットワークでは，実世界では全く関わりのない有名人などをフォローすることもあり，オンライン上での関係がより強く反映される．オンラインにおけるユーザのつながりが変わると，類似ユーザの判定もリアルタイムに反映されるものが多いため，リアルタイム性はあると言える．また，公開されているソーシャルグラフの情報を用いているため，プライバシの問題は少ないと言える．

第二に，GPS ロガーによる軌跡の解析を用いた手法では，実世界におけるユーザの位置情報を利用している．しかし，膨大な軌跡データの中から，ユーザが興味を持った場所を判定することが必要であり，多くの計算量を必要とするという問題点がある．また，類似度

を計算するたびに全ユーザの軌跡データを回収する必要があり，リアルタイムな類似ユーザ判定は不可能である．さらに，GPS ロガーの電源を入れている状態では常に位置情報が記録されてしまうため，ユーザのプライバシは全く考慮されていない．

最後に，携帯電話やスマートフォンの GPS 機能を用いて定期的に測位を行う方法では，測位した位置情報を随時サーバに送信することで，ある程度のリアルタイム性は確保できる．しかし，GPS で常に測位を行うと多くの電力を消費するため，携帯電話としての機能性を大きく損なうこととなってしまう．また，プライバシについても考慮されていないと言える．

本研究においては，発言に付加された位置情報を利用することで，実世界の行動範囲や興味を持つ場所が似ているユーザを検出することが可能である．また，発言はソーシャルメディア上からリアルタイムに取得するため，類似ユーザの判定結果を常に最新の状態に保つことが可能である．プライバシ問題においても，ユーザが自ら公開した位置情報を用いることで，ユーザの意図しない位置情報を取得してしまうことは防ぐことができる．

6.1.2 プライバシ

ユーザの位置情報を扱うアプリケーションでは，サービスの利便性とユーザのプライバシとのバランスが問題となることが多い．ユーザの現在位置はプライベートな情報であり，みだりに公開されると悪用されたり犯罪に巻き込まれてしまうおそれもある．位置情報を投稿したユーザは外出中であると言えるため，今家にいないユーザを一覧するサイトも登場し，むやみな位置情報の公開に警鐘を鳴らした [2].

このような問題に対処するため，以下のような手法が研究されている．

位置情報を分かりにくくする手法

位置情報を公開する際，緯度経度のようなピンポイントな情報ではなく，様々な加工を加えて第三者にユーザの正確な位置を把握させないようにする手法がある．具体的には，自宅に近い位置情報データを削除する手法，GPS 座標にランダムな誤差を与える手法，地図をメッシュに区切り，本当の位置が含まれるメッシュの座標のみを与える手法，特定の期間のログを削除する手法，近くにいる他人の位置情報と混在させる手法などが提案されている [15].

位置情報を公開する対象を制限する手法

位置情報をユーザが指定した特定の相手とのみ共有する手法がある．Google Latitude[8] では，ユーザが承認した友人のみに位置情報が公開される．また，このような承認ポリシーを簡略化する研究も行われている [27].

6.2 本章のまとめ

本章では，本研究と関連研究との比較を行った．位置情報処理における研究では，類似したユーザの判定手法について，既存のソーシャルグラフを用いた手法と GPS ロガーによる軌跡の解析を用いた手法について挙げた．そして，ユーザの位置情報を利用しているか，リアルタイム性があるか，プライバシに対する問題があるかについてそれぞれ比較し，本研究の利点をまとめた．最後に，位置情報共有アプリケーションにおいて重要なプライバシ問題について述べ，関連研究における手法を比較した．

第7章　結論

本章では，本論文のまとめについて述べ，本研究が提案したGLoBESの考察を行う．最後に，今後の課題と展望について述べる．

7.1 本論文のまとめ

本論文では，ソーシャルメディアにおける位置情報付きの発言を解析することで，ユーザ間の類似度を推定する手法を提案した．Twitter や Facebook などソーシャルメディアにおいてはユーザの交友関係が大きな価値を生むため，ユーザ同士の関係や趣味などが似ているユーザを発見する手法が重要である．本研究では，位置情報付きの発言から類似ユーザを推定する GLoBES (Grouping Algorithm Based on Location of Micro-Blog Entries) を構築し，実世界のユーザの行動パターンを反映した類似ユーザ推定手法を実現した．本論文では GLoBES を設計，実装し，評価を行った．GLoBES は，発言取得モジュール，発言解析モジュール，クラスタリングモジュール，類似度計算モジュール，グループ情報取得モジュールによって構成される．ソーシャルメディアの発言を解析することで類似ユーザを推定し，グループ情報として位置情報共有アプリケーションから利用可能にした．GLoBES は従来の類似ユーザ推定手法と異なり，GPS ロガーなど特殊な機器を必要としないため，既存のソーシャルネットワークにおける全世界のユーザ情報を活用することが可能である．また，推定した類似ユーザ情報は再び既存のソーシャルネットワークに対して適用可能であるという点でも有益である．

本研究の成果により，行動パターンや興味を持つ場所が似ているグループ情報がアプリケーションから利用可能になれば，グループの人数や属性に応じた情報提供や広告配信など，新たなサービスの可能性が生まれると考えられる．これはあらゆるユーザの位置情報が取得でき，誰でもどこでもネットワークにつながるようになるであろう今後の社会において，大きな利益をもたらすものである．

7.2 今後の課題

今後の課題としては，より適切な評価基準を定めること，より正確な類似ユーザ推定のために必要なパラメータを検証すること，グループ情報を用いたアプリケーションの構築の三つを挙げる．第一に，実世界の行動パターンや興味を持つ場所が似ていると定量的に評価するためのより正確な基準を設定することが求められる．現状のアンケートや既存のソーシャルグラフを用いた評価手法では，実際の行動パターンを反映した推定結果の評価としては不十分な点も残されている．

第二に，より正確な類似ユーザ推定のために，解析に使用するパラメータの調整を行うことが求められる．現在は発言の位置情報のみを使用し，3 つの粒度のクラスタに分割している．改良案としては，発言の時間帯や発言内容など，より詳細なデータを用いて解析を行うアプローチと，クラスタ粒度の分割数やそれぞれの重み付けについて適切な値を検証するアプローチが考えられる．

最後に，GLoBES を用いて計算したグループ情報を実際に活用するアプリケーションを作成することを課題として挙げる．背景となる位置情報取得機能付き端末とソーシャルメディアは今後ますます普及していくと考えられる．今後，グループ情報を用いたアプリケーションを実際に作成し，将来の社会における位置情報技術の活用に対して貢献することを目指している．

謝辞

本研究を進めるにあたり，御指導を賜りました，慶應義塾大学環境情報学部 教授 徳田 英幸博士に深く感謝致します．また，貴重な御助言を頂きました慶應義塾大学環境情報学部 准教授 高汐 一紀博士，慶應義塾大学環境情報学部 専任講師 中澤 仁博士に感謝致します．

常日頃からご指導ご鞭撻を賜り，研究方針，論文執筆について大変多くの御助言を頂きました，間 博人氏，斉藤 匡人氏，森 雅智氏，本多 倫夫氏，金澤 貴俊氏に，深く御礼申し上げます．また，半学半教の精神でお互いを高めあった，徳田・村井・楠本・中村・高汐・重近・バンミーター・植原・三次・中澤・武田合同研究プロジェクトの皆様に感謝致します．

そして，幾多の困難を共に乗り越えてきた星 北斗氏，天野 賢二氏をはじめ，同期として励まし合った Dao Thanh Chung 氏，中原 洋志氏，瀧本 拓哉氏，望月 剣氏，丹羽 亮太氏，西 和也氏，堀川 哲郎氏，上田 真央氏，Nguyen Thuy Le 氏，Vu Dinh Long 氏に感謝と尊敬の意を表します．

最後に，陰ながら今日まで支えてくれた両親に心から感謝と敬愛の意を表し，謝辞と致します．

平成 23 年 1 月 19 日
蛭田 慎也

参考文献

[1] Apple Inc. iPhone. Website, December 2010. http://www.apple.com/jp/iphone/.

[2] Barry, Borsboom and Boy, van Amstel and Frank, Groeneveld. Please Rob Me. Website, December 2010. http://pleaserobme.com/.

[3] 株式会社コロプラ. コロニーな生活☆PLUS. Website, December 2010. http://colopl.co.jp/.

[4] Justin Cranshaw, Eran Toch, Jason Hong, Aniket Kittur, and Norman Sadeh. Bridging the gap between physical location and online social networks. *Proceedings of the 12th ACM international conference on Ubiquitous computing*, 2010.

[5] 株式会社 駅探. 駅探モバイル. Website, December 2010. http://ekitan.com/k-tai/.

[6] Facebook, Inc. Facebook. Website, December 2010. http://www.facebook.com/.

[7] FOURSQUARE LABS, INC. Foursquare. Website, December 2010. http://foursquare.com/.

[8] Google, Inc. Google Latitude. Website, December 2010. http://www.google.com/intl/ja_jp/mobile/latitude/.

[9] Google, Inc. Google バズ. Website, December 2010. http://www.google.com/buzz.

[10] Google, Inc. モバイル Google マップ. Website, December 2010. http://www.google.com/mobile/maps/.

[11] 株式会社 はてな. はてなココ. Website, December 2010. http://c.hatena.ne.jp/.

[12] KDDI 株式会社. 安心ナビ. Website, December 2010. http://www.au.kddi.com/anshin/.

[13] KDDI/Powered by NAVITIME JAPAN. EZ ナビウォーク. Website, December 2010. http://eznavi.auone.jp/.

[14] Koozyt, Inc. PlaceEngine. Website, December 2010. http://www.placeengine.com/.

[15] John Krumm. A survey of computational location privacy. *Personal and Ubiquitous Computing*, Vol. 13, No. 6, pp. 391–399, October 2008.

[16] Quannan Li, Yu Zheng, Xing Xie, Yukun Chen, Wenyu Liu, and Wei-Ying Ma. Mining user similarity based on location history. *Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems*, 2008.

[17] 株式会社ライブドア. ロケタッチ. Website, December 2010. http://tou.ch/.

[18] 株式会社マピオン. ケータイ国盗り合戦. Website, December 2010. http://kntr.jp/.

[19] 株式会社エヌ・ティ・ティ・ドコモ. イマドコサーチ. Website, December 2010. http://www.nttdocomo.co.jp/service/safety/imadoco/.

[20] Open Handset Aliance. Android. Website, December 2010. http://www.openhandsetalliance.com/.

[21] PerBlue, Inc. Parallel Kingdom. Website, December 2010. http://www.parallelkingdom.com/.

[22] セコム株式会社. ココセコム. Website, December 2010. http://www.855756.com/.

[23] Skyhook, Inc. SKYHOOK. Website, December 2010. http://www.skyhookwireless.com/.

[24] ソフトバンクモバイル株式会社. 位置ナビ. Website, December 2010. http://mb.softbank.jp/mb/service/3G/ichinavi/.

[25] 総務省. 事業用電気通信設備規則. Website, December 2010. http://law.e-gov.go.jp/htmldata/S60/S60F04001000030.html.

[26] Karen P Tang, Jialiu Lin, Jason I Hong, Daniel P Siewiorek, and Norman Sadeh. Rethinking location sharing: exploring the implications of social-driven vs. purpose-driven location sharing. *Proceedings of the 12th ACM international conference on Ubiquitous computing*, 2010.

[27] Eran Toch, Justin Cranshaw, Paul Hankes Drielsma, Janice Y Tsai, Gage Kelley Patrick, Springfield James, Lorrie Cranor, Jason Hong, and Norman Sadeh. Empirical models of privacy in location sharing. *Proceedings of the 12th ACM international conference on Ubiquitous computing*, pp. 1–10, 2010.

[28] Twitter, Inc. Streaming API Documentation. Website, December 2010. http://dev.twitter.com/pages/streaming_api.

[29] Twitter, Inc. Twitter. Website, December 2010. http://twitter.com/.

[30] WILLCOM, Inc. 先進のネットワーク. Website, December 2010. http://www.willcom-inc.com/ja/service/reason/network/.

[31] 株式会社矢野総合研究所（編）. 拡大する電子地図利用・位置情報活用ビジネスの現状と展望 2009. 株式会社 矢野総合研究所, 2009.