

社会センサデータ生成・共有基盤における データフロー制御機構

愛甲 善之助¹ 谷山 雄基¹ 横山 正浩¹ 中嶋 奎介¹ 義久 智樹² 原 隆浩¹

概要：近年普及している SNS (Social Networking Service) に投稿された短文や写真から、単語の出現頻度や写真の撮影地点を解析することで実社会の情報（社会センサデータ）を取得できる。これまでの社会センサデータ生成・共有基盤では、生成された社会センサデータを別の社会センサデータの入力とする（再利用）ためには、データベースにアクセスしてデータを取得する必要がある。データが生成されると自動的に別の社会センサに入力して再利用することで、実社会の情報を早く取得できる。そこで、本研究では、筆者らがこれまでに開発してきた生成・共有基盤におけるデータフロー制御機構の設計と実装を行った。提案するフロー制御機構を用いることで、社会センサデータが生成される毎に別の社会センサへのデータの入力を自動的に行える。さらに、SNS から取得したデータを分析する社会センサ、およびその分析結果を入力データとする社会センサを実装し、提案するデータフロー制御機構の動作確認を行った。

1. はじめに

近年、Twitter や Facebook のようなマイクロブログ、Flickr や Instagram のような動画画像投稿・共有サービスにより、膨大なテキストデータや画像データが発生している。このような SNS (Social Networking Service) は様々な人々や地方自治体等に利用されている。これらのサービスにおけるユーザの投稿を解析することで、実社会を反映するデータを得られる。本報告では、SNS への投稿等を解析して実社会のデータを取得することを、社会センサによって社会をセンシングすると捉え、取得したデータを社会センサデータと呼ぶ。図 1 に社会センサデータ生成のイメージを示す。図 1 では、人・コミュニティによる SNS への投稿等を解析して社会センサデータを生成している。これまでに筆者らの研究グループでは、社会センサデータの生成、共有を目的としたプラットフォームである、S³ システム (Social Sensor Sharing System) を開発してきた [12]。S³ システムに登録された社会センサは、生成した社会センサデータを S³ システム内にあるデータベースに蓄積させており、ユーザはこのデータベースを参照して社会センサデータを共有できる。

これまでの S³ システムでは、生成した社会センサデータはデータベースに保存していたため、別の社会センサおよび S³ システム外のプログラムの入力とする（再利用す



図 1 社会センサデータ生成のイメージ

る）ためには、該当する社会センサデータをデータベースから取得する必要がある。社会センサデータが生成されたことをユーザに通知する機能もなく、社会センサデータの生成を確かめるためには、ユーザ自身がデータベースにアクセスし確認する必要がある。この操作を省略させることで、自動的に別のプログラムに入力でき、実世界の情報を早く取得できる（詳細は 3.1.2 項に記述）。

そこで本研究では、S³ システムで、生成された社会センサデータを別のプログラムの入力データとして利用するための、データフロー制御機構の設計と実装を行った。提案するデータフロー制御機構では、データフロー制御部が社会センサ間のフロー制御情報を管理し、社会センサと他のプログラムとの間の社会センサデータのデータフロー制

¹ 大阪大学大学院情報科学研究科

² 大阪大学サイバーメディアセンター

御を行う。データフロー制御部は、生成された社会センサデータを受信し、フロー制御情報に従って、送信先のプログラムへ送信する。提案する S^3 システムのためのフロー制御機構を用いることで、ユーザが設定したタイミングで別のプログラムへのデータの入力を自動的に行える。

以降、第2章で関連研究を紹介し、第3章で提案するデータフロー制御機構の設計について述べる。第4章で提案機構の実装とその具体例について述べ、第5章では提案機構に関する議論を行う。最後に第6章で論文をまとめる。

2. 関連研究

入力となるテキストデータに対し、分析を行う前に共通の前処理が行われることがある。例えば、Twitter の日本語ツイートの分析を行う場合、はじめに URL などのノイズ除去、形態素解析などの基本処理が行われた後、単語の出現頻度の計算や固有表現抽出などの応用的な処理が行われる。入力されたテキストデータに対して順番に処理を適用する方式は、NLP パイプライン方式 (NLP pipelining) と呼ばれ、様々な研究が行われている ([2]-[11])。

UIMA (Unstructured Information Management Architecture) [6] は、非構造化データを処理するフレームワークであり、構造化データとの橋渡しを行う。UIMA では、データは分析エンジンによって処理され、分析エンジン中のモジュールを組み合わせた集約型分析エンジンを用いることで、複合的な処理が可能となる。UIMA に準拠したソフトウェアが数多く提案されている ([2], [7], [10], [11])。例えば ClearTK [2] は、特徴抽出ライブラリや多様な分類器といった機械学習に関連するモジュール群を、UIMA の仕様に基づいて作成している。しかし、入出力形式が複雑なため分析エンジンのモジュール作成が難しい。

Stanford CoreNLP [8] は、構文解析や形態素解析など自然言語処理に関する様々な機能をまとめたライブラリであり、それぞれの処理を連結し、その順序に従って一括してテキストを処理できる。処理を行うプログラムはアノテータと呼ばれ、オプションを指定することで処理順序を定義できる。また、Jigg [9] は CoreNLP を日本語テキストの自然言語処理向けに拡張したもので、多様なユーザが容易に扱うことのできる NLP パイプラインを提供することを目的として設計されている。

これらのソフトウェアでは、分析プログラムを研究グループ間で共有することを目的として設計されているが、分析結果の共有は考えられていない。そのため、 S^3 システムのように複数のユーザ間で分析結果を再利用したり、別の処理の入力データとして利用できない。

3. S^3 システムのデータフロー制御機構の設計

本章では、 S^3 システムにおける社会センサデータのデータフロー制御機構の設計について述べる。

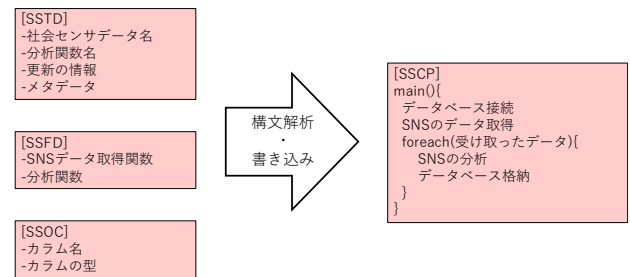


図2 従来のSSCPの作成イメージ

3.1 S^3 システムについて

S^3 システムは、社会センサデータの生成および共有を目的として設計、実装されたシステムである。まずはじめに、 S^3 システムの概要と、改善すべき課題について述べる。

3.1.1 システムへの入力とデータの使用方法

S^3 システムは、ユーザによってアップロードされたファイルをもとに、 S^3 システム内で動作する社会センサデータ生成プログラム (SSCP: Social Sensor Creation Program) を作成する。SSCP 作成のイメージを図2に示す。アップロードするファイルはそれぞれ、SNS のデータを解析するコードが記述されたファイル (SSFD: Social Sensor Function Description)、出力される社会センサデータの型定義ファイル (SSTD: Social Sensor Type Definition)、出力される社会センサデータを社会センサデータベースに格納する際の設定を記述したファイル (SSOC: Social Sensor Output Configuration) の3つのファイルからなる。 S^3 システムにおける社会センサは、入力元として S^3 システム内の SNS データベース、ユーザの手元にあるデータ、あるいは Twitter の Streaming API などにより得られるストリームデータ (連続的に発生するデータ) を、出力先として S^3 システム内の社会センサデータベースのみを設定できる。作成された社会センサは、SSTD によって定義された動作開始時刻に実行され、動作時間間隔を指定することで定期実行ができる。社会センサが実行されると、SSTD で指定された入力データを取得する関数が呼び出される。このような関数は、ユーザによって SSFD に記述されているか、 S^3 システム内であらかじめ定義されている。取得したデータを解析することで社会センサデータが生成される。生成された社会センサデータは、SSOC に定められた設定に基づいてプラットフォーム内の社会センサデータベースに格納され、ユーザ間で共有できる。生成された社会センサデータは、Web インターフェースを介して社会センサデータベースからダウンロードできる。

3.1.2 課題

これまでの S^3 システムでは、別の社会センサやプログラムで社会センサデータを再利用するために、ユーザがデータベースにアクセスして取得する必要があった。別の

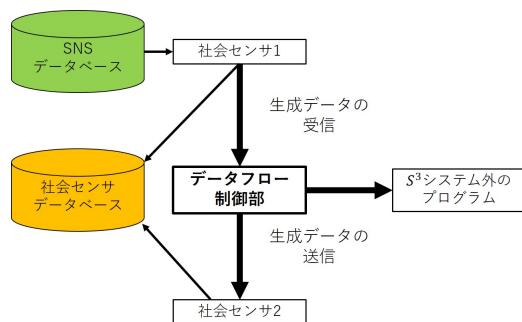


図3 データフロー制御部の動作

プログラムへのデータの入力が自動的に行うようにすることで、ユーザへの負担が減少し、実世界の情報を早く取得できる。また、データベースにアクセスすることなく再利用することで、実世界の情報をさらに早く取得できる。例えば、Twitterの投稿から話題を抽出する社会センサが生成する社会センサデータ（話題データ）を再利用して、最も注目されている話題を識別する場合、一定間隔で社会センサデータベースにアクセスして話題データの更新を確認することが考えられる。ユーザがデータベースの確認する間隔が数分程度である場合、話題が更新されてから、別の社会センサで最も注目されている話題を識別するまでに長くて数分かかることになる。データフロー制御機構を用いることで、話題が更新されると自動的に再利用して、最も注目されている話題を識別することで、早く注目話題を発見できる。

3.2 設計方針

3.1.2項で説明した問題を解決するためのデータフロー制御機構の設計方針について述べる。

3.2.1 データフロー制御機構

生成された社会センサデータを一時的に保持し、必要に応じて他の社会センサに受け渡す機能を設計する。本研究では、他の社会センサに社会センサデータを送信するために、 S^3 システムにおけるデータフロー制御部で、社会センサデータを一時的に保持し社会センサとプログラム間の通信を制御する。データフロー制御部を介した社会センサデータの流れを、図3に示す。図3では、社会センサ1が生成したデータは社会センサデータベースに保存すると共に、データフロー制御部に送信される。データフロー制御部は受信した社会センサデータを別のプログラムに送信する。社会センサ2は、データフロー制御部から受信したデータを入力として社会センサデータを生成し、社会センサデータベースに保存する。 S^3 システム外のプログラム（外部プログラム）も、データフロー制御部から受信したデータを入力として受け取り処理を行う。

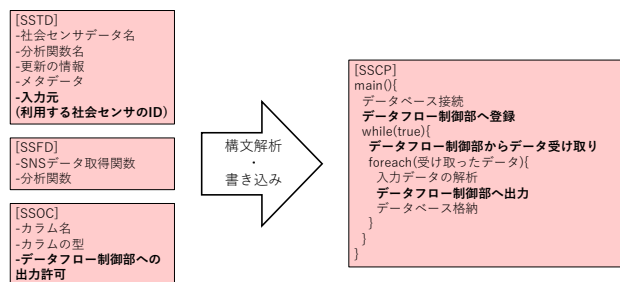


図4 データフロー制御部を介するSSCPの作成

3.2.2 S^3 システムにおけるデータの生成・格納方法の変更

データフロー制御を行うにあたり、上述したデータフロー制御部を送信先として設定できるようにし、また、データフロー制御部からデータを受信できるようにする。このために、社会センサを作成するための入力ファイルである、SSTDとSSOCの記述内容に変更を加える。さらに、これらの記述を反映させるために、SSTDとSSOCを構文解析し、構文解析結果から実際の分析を行うSSCPを作成する方法についても変更を加える。図4にデータフロー制御部を介するSSCPの作成イメージを示す。図中の、SSTDおよびSSOCの太字の部分が、提案するデータフロー制御部を用いるために追加された要求事項、SSCPの太字の部分が、提案するデータフロー制御部を用いて実現できる機能である。

また、社会センサが生成した社会センサデータや、社会センサデータベースに格納されている社会センサデータを、外部プログラムで再利用するために、外部プログラムと通信できるようにする。

3.3 構成要素

データフロー制御部は、プロセス間通信によって S^3 システムに登録された社会センサ間でのデータのやり取りを補助する。以下の要素で構成される。

社会センサデータルーティングテーブル

社会センサデータルーティングテーブルに、社会センサデータを再利用する社会センサが登録される。データフロー制御部は、社会センサデータルーティングテーブル上の登録情報を参照し、社会センサデータの振り分けを行う。

社会センサデータバッファ

社会センサデータバッファは、データフロー制御部に送信された社会センサデータ本体を、データを再利用する各社会センサへ送信するために、一時的に保持するバッファである。社会センサデータバッファでは、社会センサデータの書き込みおよび読み込みが同時に発生しないように、排他制御を行う。

社会センサデータコンテナ

表 1 社会センサデータコンテナに含まれる情報

フィールド	内容
動作要求名	データフロー制御部に要求する動作名を記述 (register, devote, reuse, delete のいずれか)
送信社会センサ ID	このコンテナを送信した社会センサの ID を記述
要求先社会センサ ID	このコンテナを送信した社会センサの 要求先社会センサの ID を記述
データ名	データ本体の名前を記述
データ本体	データ本体を記述
データ譲渡可否情報	生成したデータのほかの社会センサへの送信を 許可するか否かを記述

社会センサデータコンテナは、データフロー制御部と社会センサとの間で送受信される、社会センサデータを格納するデータコンテナである。社会センサデータコンテナには、社会センサデータのほか、データフロー制御部への要求、および要求に必要な情報が含まれる。社会センサデータコンテナに記述できるデータの種類とそれぞれの内容について、表 1 に示す。動作要求名については 3.4 節で説明する。データフロー制御部は、これらの情報を参照して、社会センサの要求ごとに適した処理を行う。データフロー制御部に要求を出す際に必要のない情報は、省略できる。社会センサの ID とは、 S^3 システムに登録された各社会センサに自動的に割り振られる識別子である。

3.4 データフロー制御部・社会センサ間の通信

表 1 で示した動作要求名に対して、データフロー制御部は動作を行う。動作要求名が“register”の場合、再利用する社会センサの登録を行う。“devote”の場合、社会センサが生成した社会センサデータの受信，“reuse”の場合、データフロー制御部が保持している社会センサデータの送信を行う。動作要求名が“delete”の場合、該当する社会センサの登録内容の削除をそれぞれ行う。

一例として、再利用するために社会センサデータを受信する場合の処理手順を図 5 に示す。送信社会センサ ID には自身の社会センサ ID を設定する。データフロー制御部は、動作要求名が“devote”のデータを受信すると、社会センサデータバッファにデータを一時的に保持するための、受信処理を開始する。まず、社会センサデータルーティングテーブルを参照し、この社会センサの生成するデータを要求する社会センサが存在するか確認する。データを要求する社会センサが存在する場合、社会センサデータバッファの、受信要求を発行した社会センサに対応する所定の領域に、一時的に保持する。このとき、データ保持領域では排他制御が行われるため、この操作が終了するまで保持するデータの読み書きは行われない。一方、データを要求する社会センサが存在しない場合は、データの受信要求を棄却する。データフロー制御部が、生成された社会センサデータを受信する際の動作イメージを図 6 に示す。図 6 では、社会センサ A が社会センサデータを生成するとデー

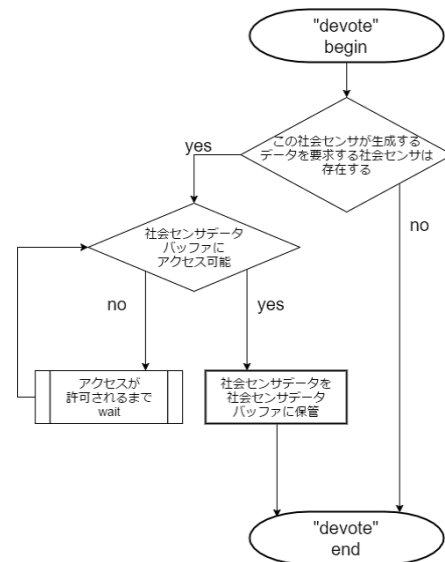


図 5 受信要求のフローチャート

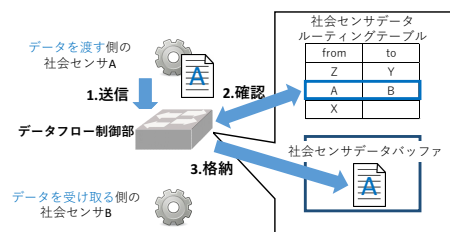


図 6 受信の要求に応じたデータフロー制御部の動作イメージ

タフロー制御部に送信する様子を表している。データフロー制御部は社会センサデータルーティングテーブルを確認し、再利用する社会センサがあれば、社会センサデータバッファに格納する。

3.5 外部プログラムとの通信

外部プログラムは、要求する社会センサデータが生成されるに社会センサデータをプログラムに送信されるように、必要とする社会センサデータや出力フォーマットを指定してデータフロー制御部に登録の要求を出す。該当する社会センサが稼働中の場合、データフロー制御部は登録の要求を受理する。 S^3 システムの社会センサが社会センサデータを生成すると、そのデータはデータフロー制御部に送られ、データフロー制御部に登録されているプログラムが要求するデータであった場合、指定されたフォーマットでそのデータをプログラムに送信する。

4. データフロー制御機構の実装

第 3 章で述べた設計に基づき、 S^3 システム上にデータフロー制御機構を実装した。本章では、実装の詳細を述べた後、 S^3 システムのデータフロー制御機構を利用する社会センサの具体例を示す。

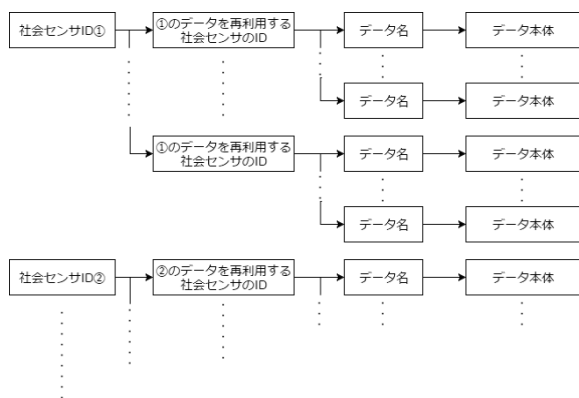


図 7 データフロー制御機構で用いた HashMap のデータ構造

表 2 クラス内変数と対応する
社会センサデータコンテナのフィールド名

変数名	フィールド
OperationName	動作要求名
ReuseSSID	送信社会センサ ID
RequiredSSID	要求先社会センサ ID
DataName	データ名
Data	データ本体
PermitFlow	データ譲渡可否情報

4.1 実装環境

社会センサルーティングテーブルおよび社会センサデータバッファの構築，社会センサからの要求の処理ロジックなど，データフロー制御部は Java1.8.0.66 で実装した。

社会センサの新規登録，社会センサの確認，社会センサデータの取得などの機能は，先行研究のものをそのまま利用した。ユーザがシステムに入力するファイルも先行研究と同様に，SSFD は Java クラスファイルもしくは JAR ファイル，SSTD および SSOC は XML ファイルとしている。

4.2 データフロー制御部の実装

本節では，3.3 節で述べた，データフロー制御部の各構成要素の実装方法について説明する。

4.2.1 社会センサデータルーティングテーブルと社会センサデータバッファの実装

社会センサデータの要求先と送信先，送信するデータの種別および社会センサデータを，Java の連想配列を用いて格納した。HashMap の構造を図 7 に示す。社会センサ ID をキー値とし，その社会センサが生成したデータを再利用する社会センサを連想配列でもち，さらに再利用するデータ名とデータ本体を保持する。

4.2.2 社会センサデータコンテナの実装

社会センサとデータフロー制御部の間で送受信させる社会センサデータコンテナは，Java のクラスとして定義した。表 2 に示すように，各フィールドに対応する変数を設定し，社会センサデータコンテナに対して読み書きするた

めのメソッドを実装した。データフロー制御部は社会センサデータコンテナを用いて，社会センサデータの送受信を行う。

4.2.3 社会センサデータの外部プログラムへの自動出力

社会センサデータが生成される度に外部のプログラムがデータを取得する場合に用いる，社会センサの自動送信機能の実装には社会センサデータが生成されると，外部サービスへ社会センサデータが図 8 のような形式で送信される。要素 row 内の各子要素名は社会センサデータの各カラム名を指しており，子要素の内容が社会センサデータである。外部プログラムへのデータの送受信には，RESTful API を用いた。RESTful API は HTTP メソッドを用いて Web システムを外部から利用するプログラムの呼び出し規約の一つであり，XML 形式や JSON 形式で呼び出しや応答を記述できる。

RESTful API を用いて社会センサデータを他の社会センサや外部プログラムへ入力する方法を図 9 に示す。外部プログラムは社会センサの検索を行うため，以下のパラメータを RESTful API で S³ システムに送信する。

- API キー（必須）：プログラミングインタフェースを利用するユーザを特定
- 出力フォーマット（必須）：JSON 形式と XML 形式のいずれかを選択
- 社会センサ ID（任意）：入力する社会センサデータの ID
- データカラム名（任意）：社会センサ ID と組み合わせで，社会センサデータの列を指定
- SQL 文（任意）：SELECT 文を記述，複数の社会センサや複数のデータカラム名を指定可能

入力された API キーが登録されたユーザの API キーと一致していれば，指定されたパラメータより社会センサデータを検索する。

4.3 具体例：ツイート分析による実世界イベント検出

本節では，実際の社会センサの例を用いて，実装したデータフロー制御機構の動作を説明する。

文献 [1] において，ツイートを意味的に類似するもの同士でクラスタリングする手法が提案されている。この手法を用いることで，ツイートのクラスタ集合に対して機械学習を用い，実世界でのイベントの発生に関連するクラスタと，それ以外のクラスタに分類できる。本研究では，データフロー制御部の動作を確認するために，イベント検出を行うプロセスを社会センサと見立て，本手法の一部を実装した。具体的には，ツイートのクラスタリングを行う社会センサと，生成されたツイートクラスタの分類を行う社会センサの，二つの社会センサに全体の処理を分割して設計・実装を行った。二つの社会センサに分けることで，別の社会センサに生成されたツイートクラスタを入力して，他の

```

<result stat="ok">
  <row>
    <lon type="java.lang.Double">135.521447</lon>
    <id type="java.lang.Long">30819858666</id>
    <focallength type="java.lang.String">10.0 mm</focallength>
    <dir type="java.lang.String"></dir>
    <f35mm type="java.lang.String">15 mm</f35mm>
    <model type="java.lang.String">NEX-6</model>
    <photo_secret type="java.lang.String">f30c120e</photo_secret>
    <farm_id type="java.lang.Integer">6</farm_id>
    <lat type="java.lang.Double">34.809588</lat>
    <server_id type="java.lang.Integer">5607</server_id>
    <post type="java.lang.Long">1478599869000</post>
    <AOV type="java.lang.String"></AOV>
    <taken type="java.lang.Long">1478500721000</taken>
  </row>
  ...
</result>

```

図 8 受信する社会センサデータ例

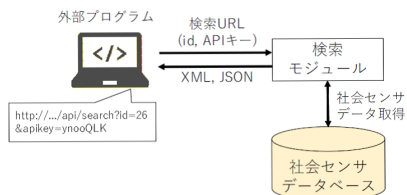


図 9 社会センサデータを他のプログラムへ入力する流れ

応用に利用できる。

4.3.1 ツイートクラスタリング

まず、Twitter から取得したツイートを処理しデータフロー制御部に社会センサデータを送信する、ツイートクラスタリングを行う社会センサについて説明する。

プログラムではまず、Twitter の Stream API から取得したツイートに対し、形態素解析を行い名詞のみを抽出する。ここでは、ツイート中の名詞を索引語としてそれぞれの出現頻度を計算し、各ツイートを単語頻度ベクトルを作成する。この際、ツイート中の URL 文字列の除去や、ハッシュタグ中に含まれる名詞の出現頻度を 2 倍に補正する、といった前処理を行っている。

次に、意味的に似た内容であるツイート同士をクラスタリングする。具体的には、あるツイートに注目したとき、既存のクラスタのうち、そのツイートとの類似度が最大となるクラスタに対し、新たにツイートを追加する。ツイートとクラスタの類似度は、クラスタが含むツイート集合の単語頻度ベクトルの重心と、注目するツイートの単語頻度ベクトルから算出されるコサイン類似度を用いる。ただし、類似度に対する閾値が設定されており、ツイート・クラスタ間の最大類似度が閾値未満である場合は、ツイートをいずれのクラスタにも追加せず、そのツイートのみのクラスタを新たに生成する。これらの一連の処理を、取得したツイートに対して行う。登録する社会センサの情報を表 3 に示す。これらの情報は全て、SSTD または SSOC に記述される。

4.3.2 ツイートクラスタの特徴抽出

次に、データフロー制御部から送信されるツイートクラスタを受信し、分類に有効な特徴量の抽出を行う社会センサについて説明する。

表 3 クラスタリングの例における社会センサ情報

社会センサデータ名	TweetClustering
更新間隔	1 時間ごと
検索用キーワード	Twitter, イベント検出
終了予定時刻	2017 年 2 月 15 日 13 時
分析内容	類似ツイートのクラスタリング
データの公開範囲	全ユーザ
プログラムの公開範囲	全ユーザ

表 4 ツイートクラスタの特徴抽出の例における社会センサ情報

社会センサデータ名	FeatureExtraction
更新間隔	データが生成され次第
検索用キーワード	Twitter, イベント検出
終了予定時刻	2017 年 2 月 15 日 0 時
分析内容	ツイートクラスタの特徴抽出
データの公開範囲	全ユーザ
プログラムの公開範囲	全ユーザ

分類に有効な特徴は、temporal feature, social feature, topical feature, twitter-centric feature の 4 種類に大きく分けられる [1]。temporal feature は、ツイートクラスタ中で出現頻度の高い単語が通常時の出現頻度とどれくらい異なるかを示し、social feature はツイートクラスタ中のリツイートやリプライ、メンションの割合である。topical feature は、ツイートクラスタ内でのトピックの一貫性の度合いであり、最後に twitter-centric feature は、ツイートに含まれるハッシュタグの複雑さを示す特徴量である。データフロー制御部からツイートクラスタを受信する毎に、クラスタ中のツイートを走査しこれらの特徴量を計算する。登録する社会センサの情報を表 4 に示す。

5. 議論

本章では、実装したデータフロー制御機構に関する評価と、今後の課題について議論を行う。

5.1 機能比較

データフロー制御に関する機能比較を表 5 に示す。生成されたデータを再利用する幾つかのフレームワークがある。提案するデータフロー制御機構を用いることで、 S^3 システムにおいて、社会センサデータを再利用できる。fluentd^{*1}は生成されたデータを受信して他のプログラムに送信して再利用できる汎用的なプログラムである。UIMA や CoreNLP は、NLP のために開発された、データを再利用する開発環境である。

S^3 システムでは、社会センサの実行中でも、データフロー制御部に登録することで、再利用するデータを登録でき、「フレームワーク実行中の再利用の登録」を行える。fluentd においても、再利用するデータをタグで指定するこ

^{*1} <http://www.fluentd.org/>

表 5 データフロー制御に関する機能比較

フレームワーク	再利用	フレームワーク 実行中の 再利用の登録	再利用元の 指定方法	再利用先
S^3 システム (提案する機構)	○	○	社会センサ ID	SSCP
fluentd	○	○	タグ	プログラム
UIMA	○	×	-	-
CoreNLP	○	×	-	-

とで, fluentd の動作中に再利用するデータを登録できる. UIMA や CoreNLP では, あらかじめモジュールのつながりでプログラムを記述して開発環境に入力するため, プログラムの実行中に再利用元を登録することはできない.

「再利用元の指定方法」とは, 再利用するデータの指定方法を示す. S^3 システムでは, 社会センサ ID を指定して再利用を登録する. fluentd では, 上記した通り, タグで再利用元を指定できる. UIMA と CoreNLP は, 再利用元を登録する機能自体がない.

「再利用先」とは, 生成されたデータを受信して再利用するモジュールを示す. S^3 システムでは, 社会センサデータを生成するプログラムである SSCP にデータフロー制御機構がデータを送信する. fluentd では, プログラム自体にデータを送信できるが, 中間処理結果となるデータはデータベースに保存されない.

5.2 従来の S^3 システムとの比較

本節では, 実装したデータフロー制御機構の評価のため, データフロー制御部における社会センサの処理時間を計測した. 実装したデータフロー制御機構では, 社会センサ間の社会センサデータの送受信をプロセス間通信によって行っているため, この分処理時間が長くなると考えられる. そこで, 同様の処理を行う単体の社会センサを実装し, データフロー制御機構を用いない場合と比較をした.

Intel Core i7 CPU@2.7GHz with 16.0GB RAM を搭載する macOS Sierra 上で動作する計算機上で実験を行った. 社会センサは, 15800 個のツイートに 500 ツイートずつ処理し, 生成または更新されたクラスタの特徴抽出を行う. 500 ツイート毎の処理時間の推移を, 図 10 に示す. 図 10 から, ツイートの処理が進むにつれて, 処理時間が増加していることがわかる. これは, ツイートの処理が進むと, ツイートクラスタ数および各ツイートクラスタが含むツイート数が大きくなることで, ツイートクラスタリングおよびツイートクラスタの特徴抽出に要する時間が長くなるためである. また, 図 10 から, データフロー制御機構を利用する社会センサの処理時間が, データフロー制御機構を利用しない社会センサの処理時間より長くなっている. これは, データフロー制御機構との通信の確立や社会センサデータコンテナの送受信に伴うものである.

データフロー制御機構を利用しない社会センサでは, 分

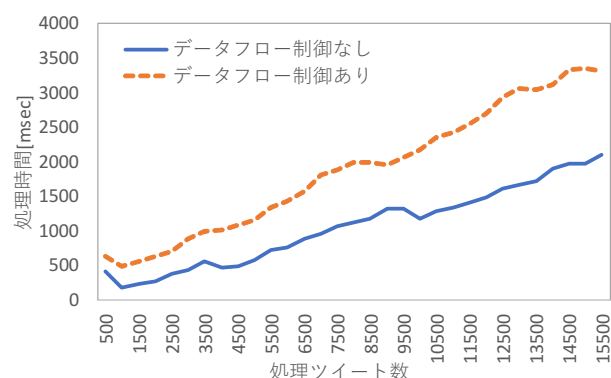


図 10 データフロー制御部の有無と処理時間

析過程で生成されるツイートクラスタが社会センサデータベースに保存されず, ユーザ間で共有できない. データフロー制御機構を利用せずに, ツイートクラスタおよびツイートクラスタの特徴量をユーザ間で共有するためには, 生成されたツイートクラスタを手動でダウンロードし, 特徴抽出を行う社会センサに入力する必要がある.

5.3 フロー制御機能の拡張

今後, 以下のデータフロー制御機構の拡張が考えられる.

5.3.1 データフロー制御部における複数のデータ要求先の指定

Twitter や Flickr といった単一の SNS から取得したデータだけを分析するのではなく, 複数 SNS のデータを同時に分析する場合が考えられる. 例えば, SNS のデータを利用した実世界イベント検出では, 4.3 節に示したような Twitter に投稿されたツイートの分析によって行われるもののほか, Flickr における位置情報付き画像データの時間的なアップロード数の変化に着目して行われるものがある.

上記の応用を実現させるため, 複数の SNS データの分析結果を入力とする社会センサを設計する場合が考えられる. この場合, データフロー制御部は複数の社会センサから受信した社会センサデータを, 単一の社会センサへ送信する必要がある. 現状では, ある社会センサデータを複数の社会センサへとマルチキャストできるが, 複数の社会センサデータを単一の社会センサへ送信する機能はない. データフロー制御部において複数のデータ要求先を指定できるよう対応することで, S^3 システムの応用を広げられる.

5.3.2 データフロー制御部における送信間隔の調整

S^3 システム上で動作する社会センサは, SSTD 内で更新間隔を指定できるため, 分析結果を得るタイミングを調整できる. 一方, データフロー制御部では, データ要求先から生成された社会センサデータを受信する毎に, 送信先の社会センサへ送信する. このため, データフロー制御部からデータを受信する社会センサの更新間隔は, 要求先の社

会センサの更新間隔に依存する。要求先の社会センサの更新間隔とは異なるタイミングで処理を行う場合が考えられる。例えば、SSFD 内に所望の更新間隔で動作するための条件分岐など、分析とは別の記述することで、要求先の社会センサの更新間隔に依存しない社会センサを実現できる。

6. まとめ

本研究では、 S^3 システムにおいて社会センサ間でデータ送受信を行うための、データフロー制御機構の設計および実装を行った。提案したデータフロー制御機構では、データフロー制御部が社会センサ間のフロー制御情報を管理しプロセス間通信によって社会センサ間の社会センサデータのデータフロー制御を行う。提案するフロー制御機構を用いることで、社会センサが生成される毎に別の社会センサへの生成されたデータの入力を自動的に行える。比較評価の結果、 S^3 システムにおいてデータフロー制御機構を用いることで、社会センサデータの再利用に適したデータフロー制御機構を提供できることが分かった。

今後の課題として、複数のデータ要求先を指定できるようにすることやデータの送信間隔を調整する機能の設計実装を考えている。

謝辞 本研究の一部は、文部科学省科学研究費補助金・基盤研究 (A) (26240013), JST 国際科学技術共同研究推進事業 (戦略的国際共同研究プログラム) の研究助成によるものである。ここに記して謝意を表す。

参考文献

- [1] Becker, H., Naaman, M. and Gravano, L.: Beyond Trending Topics: Real-World Event Identification on Twitter., *International AAAI Conference on Web and Social Media*, Vol. 11, No. 2011, pp. 438–441 (2011).
- [2] Bethard, S., Ogren, P. V. and Becker, L.: ClearTK 2.0: Design Patterns for Machine Learning in UIMA., *Language Resources and Evaluation Conference*, pp. 3289–3293 (2014).
- [3] Bontcheva, K., Derczynski, L., Funk, A., Greenwood, M. A., Maynard, D. and Aswani, N.: TwitIE: An Open-Source Information Extraction Pipeline for Microblog Text., *RANLP*, pp. 83–90 (2013).
- [4] Clarke, J., Srikumar, V., Sammons, M. and Roth, D.: An NLP Curator (or: How I Learned to Stop Worrying and Love NLP Pipelines)., *LREC*, pp. 3276–3283 (2012).
- [5] Cunningham, H., Maynard, D., Bontcheva, K. and Tablan, V.: GATE: an architecture for development of robust HLT applications, *Proc. of meeting on association for computational linguistics*, pp. 168–175 (2002).
- [6] Ferrucci, D. and Lally, A.: UIMA: an architectural approach to unstructured information processing in the corporate research environment, *Natural Language Engineering*, Vol. 10, No. 3-4, pp. 327–348 (2004).
- [7] Hahn, U., Buyko, E., Tomanek, K., Piao, S., McNaught, J., Tsuruoka, Y. and Ananiadou, S.: An annotation type system for a data-driven NLP pipeline, *Proc. of the Linguistic Annotation Workshop*, pp. 33–40 (2007).
- [8] Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J. R., Bethard, S. and McClosky, D.: The Stanford CoreNLP Natural Language Processing Toolkit., *ACL (System Demonstrations)*, pp. 55–60 (2014).
- [9] Noji, H. and Miyao, Y.: Jigg: a framework for an easy natural language processing pipeline, *Proc. of Association of Computational Linguistics*, pp. 103–108 (2016).
- [10] 砂山渡, 高間康史, 西原陽子, 徳永秀和, 串間宗夫, 阿部秀尚, 梶並知記: テキストデータマイニングのための統合環境 TETDM の開発, 人工知能学会論文誌, Vol. 28, No. 1, pp. 1–12 (2013).
- [11] 狩野芳伸, 辻井潤一ほか: UIMA を基盤とする相互運用性の向上と自動組み合わせ比較-国際共同プロジェクト U-Compare, 情報処理学会研究報告自然言語処理 (NL), Vol. 2008, No. 67 (2008-NL-186), pp. 37–42 (2008).
- [12] 中嶋奎介, 谷山雄基, 横山正浩, 義久智樹, 原隆浩, 西尾章治郎ほか: 社会センサデータ生成・共有基盤システムの設計と実装, マルチメディア, 分散協調とモバイルシンポジウム 2016 論文集, Vol. 2016, pp. 1215–1222 (2016).