DEIM 2020 E21-1

リアルタイム動画動作識別の実現に向けた分散ストリーム処理基盤の検討

高崎智香子† 竹房あつ子†† 中田 秀基††† 小口 正人†

† お茶の水女子大学 〒 112–8610 東京都文京区大塚 2–1–1
†† 国立情報学研究所 〒 101–8430 東京都千代田区一ツ橋 2–1–2
††† 産業技術総合研究所 〒 305–8560 茨城県つくば市梅園 1–1–1
E-mail: †chikako@ogl.is.ocha.ac.jp, ††takefusa@nii.ac.jp, †††hide-nakada@aist.go.jp †oguchi@is.ocha.ac.jp

あらまし 子供やお年寄りの見守りサービスや防犯を目的として家庭のセンサで取得した動画像をリアルタイムに機械学習で解析するには，データ量と解析計算量が課題となる．我々は，センサ側で姿勢推定ライブラリ OpenPose を使用して動画像から関節の特徴量データを抽出してクラウドへ転送し，クラウドでその特徴量データのみを用いて機械学習による動作識別を行うことで，処理遅延やプライバシの問題に対処するセンサとクラウドでの分散処理手法を提案している．しかし，複数家庭のセンサから連続的に送られる大量のデータをクラウドで処理するには，急激なデータの増加によるシステム負荷上昇に耐えうる処理基盤が必要である．本研究では，大量のデータを効率よく処理可能な分散ストリーム処理基盤の構築を目指して，エッジで抽出した関節の特徴量データを Apache Kafka を用いて収集し，クラウドにおいて Apache Flink の分散ストリーム処理機能を用いて機械学習を行うシステムを構築し，その性能特性を調査した．実験結果から，Flink の並列処理機能を用いることでスケーラブルに機械学習の推論を行えることが分かった．また，データ転送時のバッファリング待ち時間を制御する Flink の BufferTimeout パラメータを調節することで，解析効率が改善することが分かった．

キーワード 分散処理，機械学習，深層学習，OpenPose

A Study on a Distributed Stream Processing System for Real-Time Video Action Recognition

Chikako TAKASAKI†, Atsuko TAKEFUSA††, Hidemoto NAKADA†††, and Masato OGUCHI†

† Ochanomizu University 2-1-1 Otsuka,Bunkyo-Ku, Tokyo 112–8610, Japan
†† National Institute of Informatics 2-1-1 Hitotsubashi, Chiyoda-ku, Tokyo, 101-8430, Japan
††† National Institute of Advanced Industrial Science and Technology (AIST) 1-1-1 Umezono, Tsukuba, Ibaraki 305–8560, Japan
E-mail: †chikako@ogl.is.ocha.ac.jp, ††takefusa@nii.ac.jp, †††hide-nakada@aist.go.jp †oguchi@is.ocha.ac.jp

1. はじめに

センサ機器やクラウドコンピューティングの普及により，一般家庭でライフログを取得，蓄積し，子供やお年寄り，ペットの見守りサービスや防犯対策，セキュリティに活用されるようになってきた．M. Mohammad ら [1] は，スマートホームやスマートシティ，ヘルスケア，農業分野などの Internet of Things (IoT) デバイス (センサ) から生成される大量のストリームデータを機械学習で分析する様々な分野のアプリケーションを紹介し，大規模ストリームデータの分析とアプリケーションの目的の達成には機械学習を用いることが有望だと述べている．家庭のセンサで取得した動画像をリアルタイムに機械学習を用いて解析するには，データサイズと解析計算量が大規模であるため高性能なサーバやストレージが必要となり，それらをセンサを配備している環境に設置するのは難しい．クラウドでは潤沢な計算資源を利用することができるが，センサとクラウド間のネットワーク帯域が限られており，アプリケーションが期待する処理遅延で解析を行うのは非常に困難である．そのため，計算の一部をセンサ側，もしくはセンサ側に近いエッジデバイスで処理をするエッジコンピューティング [2] やフォグコンピュー

ティング [3] と呼ばれる分散処理が有効である．

センサ側で前処理を行う利点は，処理遅延の他にもプライバシの確保，通信コストの削減の点も挙げられる．家庭内で撮影された個人が特定できるような動画像をそのままクラウドで収集，蓄積すると，アプリケーション利用者のプライバシを侵害する恐れがある．また，センサとクラウド間の通信にモバイル網を利用している場合は，転送量に従って課金されるため，転送量の削減が求められる．よって，センサ側での前処理で動画像から特徴量を抽出してデータ量の削減と匿名化をした後，クラウドでその特徴量データを収集して解析することでこれらの問題に対処できる．我々は，センサ側で姿勢推定ライブラリ OpenPose [4] [5] [6] [7] を使用して動画像から関節の特徴量データを抽出し，クラウドでその特徴量データのみを用いて機械学習による動作識別を行うことで，処理遅延やプライバシの問題に対処する分散処理手法を提案した [8] [9]．評価から，センサからクラウドへのデータ転送量を大幅に削減できること，動画像データの情報量が大幅に失われてしまうものの特徴量のみを用いた学習でも十分な精度が得られることを確認した．

実際の利用環境では，複数家庭のセンサから連続的に送られる大量のデータをクラウドで処理するため，急激なデータの増加によるシステム負荷上昇に耐えうる処理基盤が必要である．センサから得られる大量のデータを効率良く収集，解析するには，クラウドで分散ストリーム処理基盤を用いることが有効である．分散ストリーム処理は近年多く研究されているが，このような処理基盤上では一般に収集やフィルタのような軽量な処理のみ行われている．クラウドでより高度な解析を行うためには機械学習などの高負荷なタスクの処理が必要だが，このようなタスクを用いたリアルタイム分散ストリーム処理の実現可能性は明らかでない．

本研究では，大量のストリームデータを効率良く処理可能な処理基盤の構築を目指し，分散ストリーム機械学習処理システムの性能特性を調査する．評価のため，センサで抽出した関節の特徴量データを分散メッセージングシステム Apache Kafka(以降，Kafka と呼ぶ) [10] で収集し，クラウドにおいて分散ストリーム処理フレームワーク Apache Flink(以降，Flink と呼ぶ) [11] を用いて機械学習による動作識別処理を行う分散ストリーム処理基盤を構築する．構築した処理基盤を用いて，解析スループットと総遅延時間を測定し，Flink の並列度に対するスケーラビリティについて調査した．また，Flink のデータ転送のバッファリング待ち時間を制御する BufferTimeout パラメータがスループットに与える影響を調査した．実験から，Flink の並列処理機能を用いることで機械学習の推論による解析をスケーラブルに行えることを確認した．また，解析に時間がかかる LSTM の推論を行う場合には，バッファリングせずにデータを転送し，解析に時間がかからない NN の推論を行う場合には，バッファリングしてまとめてデータを転送することで解析効率が改善することが分かった．

2. センサ，クラウド間分散動画像解析システムの概要

本研究では，図 1 のようなセンサ，クラウド間分散動画像解析システムを想定している．各一般家庭に設置されたセンサのカメラで動画像を取得し，センサ端末内で前処理を行った後，メッセージングシステムを用いてクラウドにデータを収集して分散ストリーム処理基盤上で分散機械学習を行う．メッセージングシステムには，既発表研究 [12] で高スループットで画像データの収集ができることを確認している Kafka を用いる．また，分散ストリーム処理基盤には，低遅延で処理可能であることを確認している Flink を用いる [13]．図 1 では，センサ端末で OpenPose を用いてキーポイントの座標データを抽出し，Kafka Producer から Kafka Broker に転送する．クラウドにおいて Kafka Consumer を用いて Kafka Broker からデータを受け取り，Flink の分散ストリーム処理機能を用いて機械学習の推論を行うことで動画像に含まれる動作を識別する．その後，解析結果をユーザに通知する必要があるため，Kafka Producer を用いて Kafka Broker に解析結果を転送する．クラウド側では動画像や静止画を用いず，センサ側で抽出したキーポイントデータのみを使用して解析を行う．OpenPose，Kafka，Flink について以降で説明する．

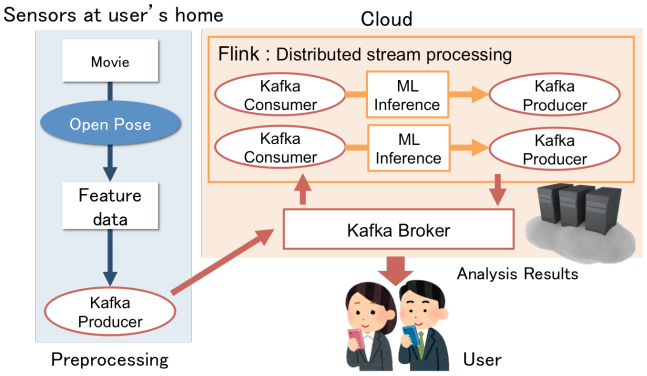

図 1　提案する動画像解析システム．

2.1 OpenPose

OpenPose は，深層学習を用いて人の関節等のキーポイント情報をリアルタイムに抽出する姿勢推定ライブラリで，カーネギーメロン大学などによって開発された．動画像や画像に含まれる人物の身体，顔，手の 135 のキーポイントを検出することが可能である．加速度センサなどの特殊センサを使わずに，カメラによる画像や動画像のみで解析できることが特徴である．また，GPU を使用することで，画像や動画像に複数の人が含まれている場合でもリアルタイムに解析できる．

2.2 Apache Kafka

Kafka は高スループットでリアルタイムに大容量データを収集，配信することを目的として開発された分散メッセージングシステムである．データを保存する Broker，Broker にデータを配信する Producer，Broker からデータを参照する Consumer という 3 つのコンポーネントで構成される．Publish-Subscribe モデルを採用し，スループットを調節することが可能である．大量のメッセージを高速に処理することが可能であり，レプリケーションを行うことで耐障害性や高可用性を実現し，リアルタイムデータパイプラインやストリーミングアプリケーションの構築に広く使われている．

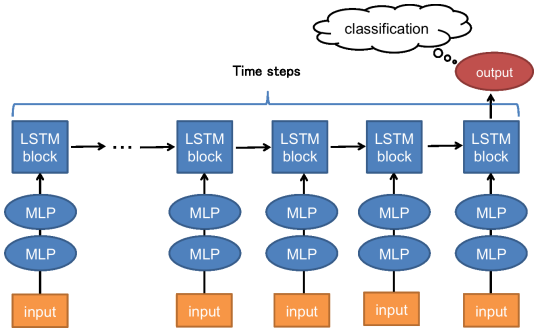

図 2　LSTM モデルの構成.

2.3　Apache Flink

Flink は，高スループット・低レイテンシを実現する分散ストリームとバッチデータ処理のためのフレームワークである．各処理をステートフルに扱うことで，障害発生時に処理を自動で復旧させる機能を持つため，耐障害性に優れている．無限ストリームを扱うデータストリーム API，静的データを扱うデータセット API，SQL のようなデータベース表現を扱うテーブル API を持つ．また，イベントストリームからパターンを検出するライブラリである FlinkCEP，機械学習ライブラリである FlinkML，グラフ API である Gelly を持ち，様々なユースケースに対応できる．

3.　本研究で使用する機械学習手法

OpenPose を用いて動画像の各フレームから抽出したキーポイントの座標データを使用し，機械学習による動作識別モデルを作成する．動作識別モデルは予め TensorFlow を用いて作成，学習し，Flink プログラムで読み込んで推論処理に用いる．データセットには，日常の動作 100 カテゴリの動画像を約 1000 ずつ集めた STAIR Actions [14] の動画像を利用する．

3.1　機械学習手法

実験では，以下の 2 つの手法で動作識別モデルを作成し，動画像に含まれる動作を識別した．各動画から 10 枚の静止画を取得し，OpenPose で抽出したキーポイントを入力として，動画像に含まれる動作のカテゴリ分類を行う．

1)　Neural Network (NN)
2)　Long Short-Term Memory (LSTM)

1) NN は人の神経細胞を模したモデルであり，完全結合の NN(MLP) を用いた．NN モデルには，入力層，中間層 3 層，出力層から構成される MLP を用い，動作のカテゴリ分類を行う．2) LSTM は，Recurrent Neural Network(RNN) という再帰型 NN モデルを拡張した手法である．RNN は，前の時間に計算された情報を記憶しておき，後の計算でこれらの情報を使用して学習を行うことで文章など連続的な情報を学習できるが，長期記憶ができないという欠点がある．LSTM はこの欠点を解消し，データの長期依存を学習可能にした．LSTM モデルの構成を図 2 に示す．各画像から取得した特徴量を 2 層の MLP で学習した後，その結果を LSTM に時間ステップごとの入力として与え，最後のステップの出力を用いて動作のカテゴリ分類を行う．

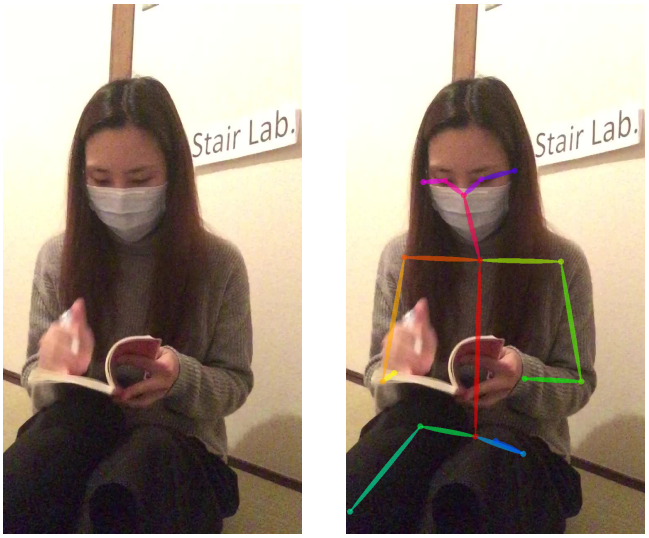

図 3　OpenPose によって取得したキーポイント.


```
"people":[{"pose_keypoints_2d":[
283.201,461.222,0.818301,321.728,618.22,
0.751273,140.097,611.302,0.643794,
87.7155,911.437,0.400931,115.677,890.546,
0.379135,517.083,621.779,0.651862,
565.945,911.436,0.742212,429.756,904.388,
0.770721,311.119,1044.03,0.349768,
178.451,1019.65,0.359588,9.12998,1225.6,
0.103076,0,0,0,426.329,1085.95,0.265937,
360.019,1054.57,0.412622,0,0,0,237.761,
426.189,0.885139,325.128,422.815,
0.826925,185.5,429.66,0.365905,405.442,
401.993,0.704888,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0]}]
```


図 4　OpenPose によって取得した座標値の一部.

3.2　使用データ

STAIR Actions データセットの各動画像から，0.3 秒ごとの等間隔に 10 枚の静止画を取得した．その後，各静止画に対して OpenPose を用いて 25 のキーポイントの画像上の x，y 座標を取得して特徴量 50 のデータを取得し，データセットを作成した．各静止画の 50 のキーポイントを時系列順に並べて，合計 500 の特徴量を 1 入力データとして使用した．OpenPose によって抽出したキーポイントの例を図 3 に，この画像から取得した座標値データの一部を図 4 に示す．データ数は 96807 で，このうちの 7 割を学習に，3 割をバリデーションに使用して動作識別モデルを作成した．

4.　実　　験

提案する分散ストリーム機械学習処理システムの解析処理性能とリアルタイム性を調査するため，処理スループットと，センサからデータを送信してクラウドで解析し，その結果を取得するまでの総遅延時間を測定した．分散ストリーム機械学習処

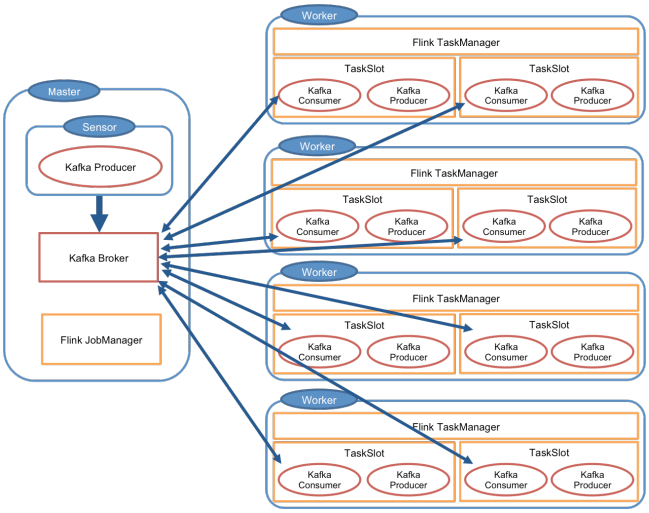

図 5　実験構成．

理システムの基本性能とクラウドでの解析手法の違いによる性能への影響を調査するため，クラウドでの処理（Flink タスク）は以下の 3 パターンで比較した．

(1)　データを受信後，解析を行わないで模擬結果を送信する

(2)　データを受信後，NN で推論して結果を送信する

(3)　データを受信後，LSTM で推論して結果を送信する

次に，Flink でのストリーム処理を効率良く行い，解析スループットを最大化するため，データ転送時のバッファリング機能を制御する BufferTimeout パラメータを以下の 3 パターンで変化させてスループットを測定した．

(a)　デフォルト (BufferTimeout 100ms)

(b)　BufferTimeout 設定なし (待ち時間なし)

(c)　BufferTimeout 5ms

Flink のストリーム処理では，ネットワーク上で 1 データずつ転送せずバッファリングされる．BufferTimeout パラメータを用いて，バッファが埋まるまでの最大待ち時間をミリ秒単位で設定できる．一般的に，スループットを最大化するには待ち時間を設定せずにバッファが埋まってから転送し，レイテンシを最小化するには待ち時間を短くしてデータを溜めずに転送する．センサプログラムから転送する各特徴量のデータサイズは 2KB で，処理データの総数は 10000 とした．

4.1　実験環境

図 5 に示す構成で，5 つのノードを用いて実験を行う．Master ノードでは，複数のセンサ端末を模擬したセンサプログラムである Kafka Producer と，Kafka Broker，Flink JobManager，Apache Zookeeper を起動する．Flink JobManager は，並列度に応じて 4 台の Worker ノードで動作している Flink TaskManager のスロットにタスクを分配する．各 Flink タスクでは，Kafka Consumer を動作させて Kafka Broker からデータを受け取り，特徴量データを用いた推論を行って動作識別を行う．その後，Kafka Producer を各 Flink タスクで動作させ，動作識別結果を Kafka Broker に転送する．データのカテゴリを生成する Kafka Topic においてデータの Partition 数を設定することで，複数のスロットにデータを分散し，並列処理を行うことができる．Partition 数は 1，20，40，80 に，各 TaskManager のスロット数を 20 として Flink の並列度は 1，20，40，80 に設定した．

表 1　実験で使用した計算ノードの性能とソフトウェアバージョン．

CPU	Intel(R) Xeon(R) CPU E5-2660 v4 @ 2.00GHz 14Cores 28Threads × 2
Memory	125Gbyte
ホスト OS	Ubuntu 16.04
コンテナ OS	Alpine Linux 3.10.1
Kafka Version	2.2.0
Flink Version	1.7.2
ZooKeeper Version (Master のみ)	3.4.14

実験に用いた計算機の性能とソフトウェアバージョンを表 1 に示す．各ソフトウェアは国立情報学研究所が提供する「学認クラウドオンデマンド構築サービス」[15] [16] を用いて Docker コンテナで配備した．Master および全ての Worker には同質のノードを用いている．Kafka はバージョン 2.2.0，Flink はバージョン 1.7.2 を使用した．本実験では，Kafka Broker は 1 ノード構成とし，レプリケーションなしとした．センサプログラム，Flink タスクプログラムは Java で実装した．

4.2　実験結果

まず，解析パターンごとにスループットと総遅延時間の関係について調査した．Flink タスクで (1) 解析を行わない場合，(2)NN の推論を行う場合，(3)LSTM の推論を行う場合のスループットと総遅延時間をそれぞれ図 6，図 7，図 8 に示す．実験は全て 5 回ずつ行い，その平均値をグラフに示している．左図は Kafka の Partition 数と Flink の並列度ごとの平均合計スループットを示し，横軸が Partition 数を，縦軸が 1 秒間に処理したデータ数を表す．右図は処理開始から 20%までのデータの平均総遅延時間を示し，横軸が Partition 数を，縦軸が総遅延時間を秒単位で表す．Flink Parallelism は並列処理を行う TaskSlot 数を表し，Flink での解析処理を分散して同時実行する．1 データの推論時間は，NN は平均 0.508ms，LSTM は平均 1.835ms であった．

図 6 から，並列度の増加によってスループットが向上していないことが分かる．(1) 解析を行わない場合は，TaskSlot における処理に時間がかからないため，Kafka Producer からのデータ転送と Partition の増加によるオーバヘッドが顕在化している．(2)NN の推論を行う場合と (3)LSTM の推論を行う場合は，図 7，図 8 から Flink の並列度を増やすことでスループットが向上しており，スケーラブルに解析できていることが分かる．一方，並列度を 80 まで増やしてもスループットがあまり向上しなかった．

実験では，センサプログラムからのデータ転送レートが Flink での処理レートを上回り，実験後半にタスクが滞留して総遅延時間が長くなってしまったため，実験開始から 20 %までのタスクの平均値で比較する．解析時間にある程度時間がかかる (3)LSTM の推論を行う場合には，(1) 解析を行わない場合，(2)NN の推論を行う場合と比較して総遅延時間が長くなってい

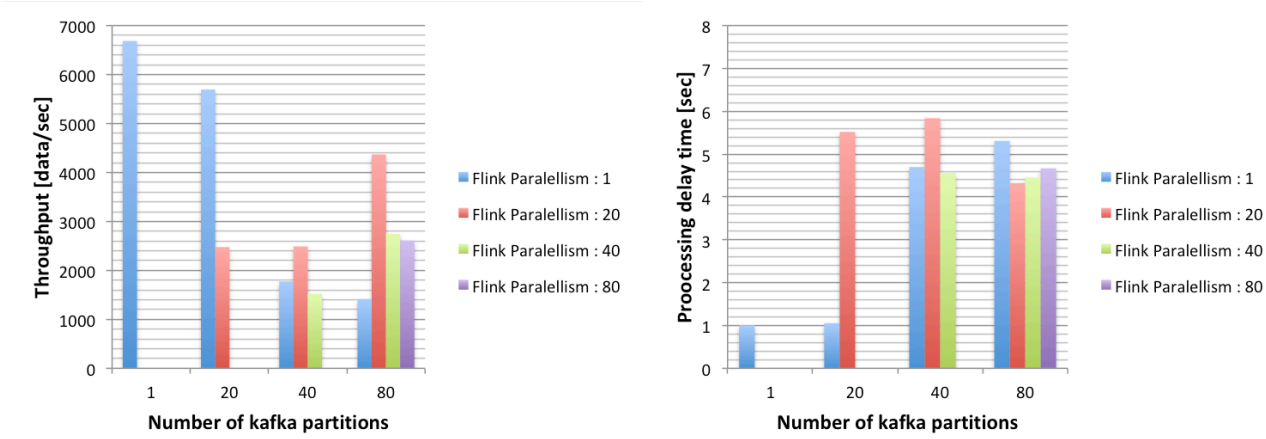

図 6　(1)Flink タスクで解析を行わない場合のスループット (左図) と開始から 20%までのタスクの平均総遅延時間 (右図).

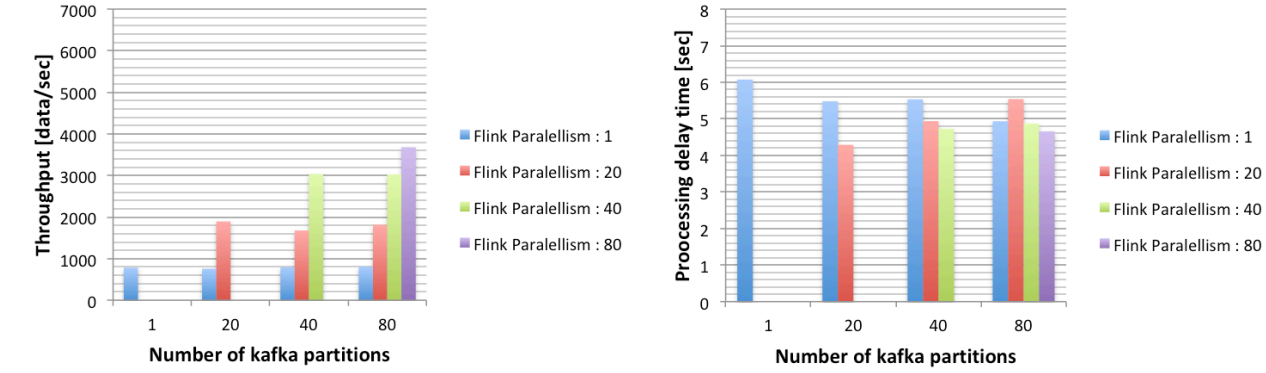

図 7　(2)Flink タスクで NN の推論を行う場合のスループット (左図) と開始から 20%までのタスクの平均総遅延時間 (右図).

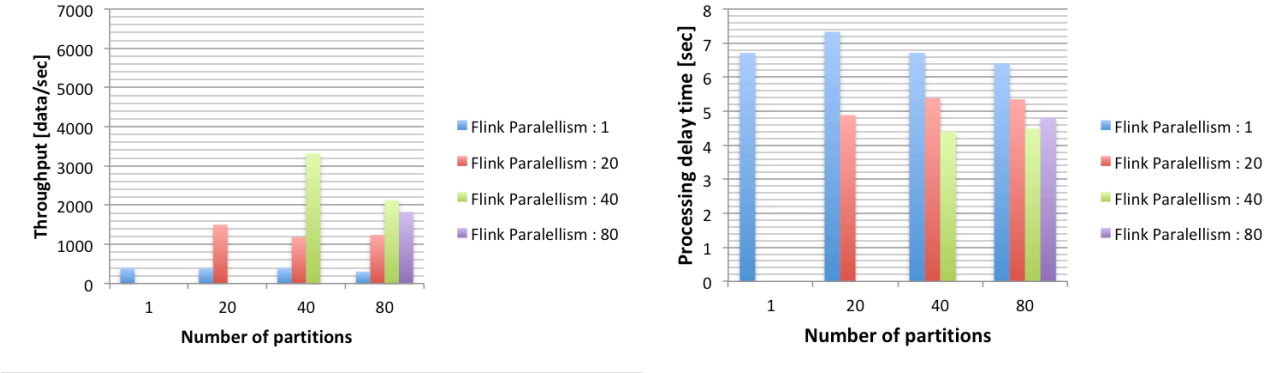

図 8　(3)Flink タスクで LSTM の推論を行う場合のスループット (左図) と開始から 20%までのタスクの平均総遅延時間 (右図).

ることがわかる．また，解析を行う場合はいずれも並列度を上げると総遅延時間が短くなる傾向が見られる．これは並列化によりタスク滞留が緩和されているためだと考えられる．

次に，Partition 数と Flink の並列度をそれぞれ 1 に設定してスループットを測定し，BufferTimeout パラメータのスループットに与える効果を調査した．解析パターンと BufferTimeout 設定パターンごとの 5 回平均スループットを図 9 に示す．横軸は解析パターンと BufferTimeout の設定パターンを，縦軸は 1 秒間に処理したデータ数を表す．

(b)BufferTimeout を設定しない場合と (c)BufferTimeout を 5ms に設定する場合を比較すると，(1) 解析を行わない場合と (2)NN の推論を行う場合には，待ち時間を設定せず，バッファに溜めてから転送したほうがスループットが高くなっている．(3)LSTM の推論を行う場合には，バッファに溜めずデータの解析終了後すぐに転送したほうがスループットが高くなっている．LSTM の推論は処理時間が長く，多少のオーバーヘッドがあっても逐一処理することで各タスクの遅延時間を削減できるため，BufferTimeout パラメータを短く設定することで効率良

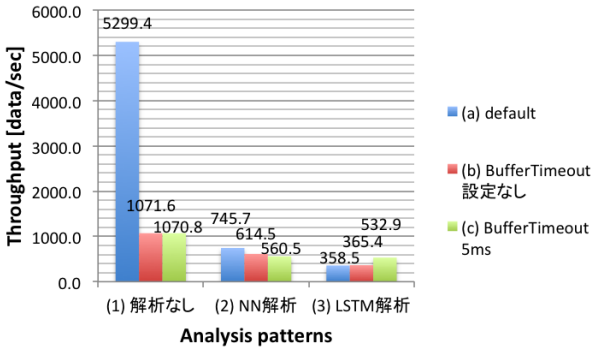

図 9　解析パターンと BufferTimeout の設定ごとのスループット．

く解析できる．それに対し，解析を行わない場合や NN のように処理時間が短い場合には，データをまとめて転送することで転送オーバーヘッドを削減できるため，BufferTimeout パラメータを設定せずバッファリングした方が効率良く解析できると考えられる．

5. 関 連 研 究

深層学習を用いた動画像の動作識別は，近年数多く研究されており，CNN や LSTM など様々な手法を用いることでより複雑な動作を高精度で識別することが可能になっている．Hara ら [17] は，動画像を入力として行動ラベルを識別するという課題に対し，2 次元の空間に 1 次元の時間空間を加えた 3 次元空間で畳み込みを行う，3D CNN ベースの様々な手法を用いた行動識別について調査した．また，Residual Network(ResNet) [18] ベースの 3D CNN を用いた行動識別による性能改善を示している．Pigou ら [19] は，動画像の一時的な情報を考慮するために特徴量を pooling して使用し，再帰と Temporal Convolution の組み合わせによる性能改善を示した．Li [20] らは，RFID で取得したデータを用いた CNN による human action のマルチクラス分類を行った．Fragkiadaki ら [21] は，LSTM の前後にエンコーダとデコーダを組み込んだモデルを使用して human pose のモーションキャプチャを生成し動作の識別，予測を行った．Ordóñez ら [22] は，CNN と LSTM の組み合わせにより加速度計，ジャイロスコープ，磁力計のデータを前処理せず融合することで識別を行った．しかし，このような処理は計算量が多く，各家庭で深層学習を用いた解析を行うのは非常に困難である．

IoT デバイスから取得したデータを用いて解析を行うためのエッジコンピューティングやフォグコンピューティングによる様々な分散処理手法が研究されている．Li ら [23] は，IoT デバイスから取得したセンサデータを用いてディープラーニングによって解析を行う手法を提案した．エッジノードにおける処理能力は限られているため，エッジコンピューティングを使用して IoT の深層学習 アプリケーションを構築し，パフォーマンスを最適化するためのオフロード戦略の設計や評価を行った．Tang ら [24] は，将来のスマートシティに向けて，膨大なインフラの統合をサポートする階層的な分散フォグコンピューティングアーキテクチャを提案した．Yang [25] は，4 つの典型的なフォグコンピューティングシステムのモデルとアーキテクチャの調査を行い，システム，データ，人間，最適化の 4 次元についてのデザインスペースを分析した．このように，センサを用いて取得したデータをクラウドで収集，解析する分散ストリーム処理によって効率良く，大量のデータを解析することが可能になっている．

分散ストリーム処理基盤では，Apache Storm，Apache Flink，Apache Spark Streaming が代表的である．Chintapalli ら [26] は，これら 3 つのストリーム処理基盤を用いて広告データの収集を行う際のスループットと処理遅延について調査し，Storm と Flink を用いて低遅延なストリーム処理を行えることを示した．Karimov ら [27] は，スマートフォンのゲーム課金や広告データを用いて Storm，Flink，Spark Streaming で分散ストリーム処理基盤を構築，評価し，Flink が低遅延，高スループットなストリーム処理基盤として優れていることを示した．しかし，このような分散ストリーム処理基盤では主に収集やフィルタのような軽量な処理のみが行われており，機械学習の推論のような複雑な解析を行う際の処理性能は明らかでない．

本研究はエッジとクラウドで処理を分散させる事によって深層学習を用いた解析をリアルタイムに行い，ストリーム処理基盤上で機械学習の推論を用いて解析するという点で異なる．

6. まとめと今後の予定

本稿では，STAIR Actions データセットの動画像から取得した画像から OpenPose を用いて抽出したキーポイントデータを Kafka で収集し，クラウドにおいて Flink を用いて機械学習による動作識別処理を行うシステムを構築し，解析処理性能を調査した．実験から，並列処理によってスケーラブルに解析を行うことができることを確認した．また，BufferTimoue 設定の調節により，解析に時間がかかる場合にはデータ転送の遅延時間を小さくし，解析に時間がかからない場合にはバッファリングしてデータをまとめて転送することで，高効率に処理できることが分かった．

今後は，家庭に配備可能なエッジデバイスとクラウド環境でのスループットや遅延時間について調査し，リアルタイム解析の実現を目指す．

謝　　辞

この成果の一部は，JSPS 科研費 JP19H04089 及び，2020 年度国立情報学研究所公募型共同研究（20S0501）の助成を受けたものです．

文　　献

[1] Mehdi Mohammadi, Ala Al-Fuqaha, Sameh Sorour, and Mohsen Guizani. Deep learning for iot big data and streaming analytics: A survey. *IEEE Communications Surveys & Tutorials*, Vol. 20, No. 4, pp. 2923–2960, 2018.

[2] Pedro Garcia Lopez, Alberto Montresor, Dick Epema, Anwitaman Datta, Teruo Higashino, Adriana Iamnitchi, Marinho Barcellos, Pascal Felber, and Etienne Riviere. Edge-centric computing: Vision and challenges. *ACM SIGCOMM Computer Communication Review*, Vol. 45, No. 5, pp. 37–42, 2015.

[3] Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. Fog computing and its role in the internet of things.

In *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, pp. 13–16. ACM, 2012.

[4] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2019.

[5] Tomas Simon, Hanbyul Joo, Iain Matthews, and Yaser Sheikh. Hand keypoint detection in single images using multiview bootstrapping. In *Proc. IEEE conference on Computer Vision and Pattern Recognition*, pp. 1145–1153, 2017.

[6] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proc. the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7291–7299, 2017.

[7] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *Proc. the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4724–4732, 2016.

[8] C. Takasaki, A. Takefusa, H. Nakada, and M. Oguchi. A study of action recognition using pose data toward distributed processing over edge and cloud. In *Proc. the 11th IEEE International Conference on Cloud Computing Technology and Science (CloudCom2019)*, pp. 111–118, 2019.

[9] C. Takasaki, A. Takefusa, H. Nakada, and M. Oguchi. Action recognition using pose data in a distributed environment over the edge and cloud. *IEICE Transactions on Information and Systems Special Section on Data Engineering and Information Management*, Vol. E104-D, No. 05, 2021（印刷中）.

[10] Apache Kafka : A Distributed Streaming Platform. . `https://kafka.apache.org/`.

[11] Apache Flink : Stateful Computations over Data Streams. `https://flink.apache.org/`.

[12] A. Ichinose and A. Takefusa and H. Nakada and M. Oguchi. A Study of a Video Analysis Framework Using Kafka and Spark Streaming. In *Proc. Second Workshop on Real-time & Stream Analytics in IEEE Big Data*, pp. 2396–2401, 12 2017.

[13] 竹房あつ子, 孫静涛, 藤原一毅, 長久勝, 吉田浩, 政谷好伸, 合田憲人. Sinet 広域データ収集基盤を用いたオンラインビデオ処理実証実験. 情報処理学会論文誌デジタルプラクティス（TDP）, Vol. 1, No. 1, pp. 45–57, oct 2020.

[14] Yuya Yoshikawa, Jiaqing Lin, and Akikazu Takeuchi. Stair actions: A video dataset of everyday home actions. *arXiv preprint arXiv:1804.04326*, 2018.

[15] 学認クラウドオンデマンド構築サービス. `https://cloud.gakunin.jp/ocs/`.

[16] A. Takefusa, S. Yokoyama, Y. Masatani, T. Tanjo, K. Saga, M. Nagaku, and K. Aida. Virtual Cloud Service System for Building Effective Inter-Cloud Applications. In *Proc. IEEE CloudCom2017*, pp. 296–303, 12 2017.

[17] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *Proc. the IEEE conference on Computer Vision and Pattern Recognition*, pp. 6546–6555, 2018.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[19] Lionel Pigou, Aäron Van Den Oord, Sander Dieleman, Mieke Van Herreweghe, and Joni Dambre. Beyond temporal pooling: Recurrence and temporal convolutions for gesture recognition in video. *International Journal of Computer Vision*, Vol. 126, No. 2-4, pp. 430–439, 2018.

[20] Xinyu Li, Yanyi Zhang, Ivan Marsic, Aleksandra Sarcevic, and Randall S Burd. Deep learning for rfid-based activity recognition. In *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems CD-ROM*, pp. 164–175. ACM, 2016.

[21] Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. Recurrent network models for human dynamics. *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 4346–4354, 2015.

[22] Francisco Ordóñez and Daniel Roggen. Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition. *Sensors*, Vol. 16, No. 1, p. 115, 2016.

[23] He Li, Kaoru Ota, and Mianxiong Dong. Learning iot in edge: Deep learning for the internet of things with edge computing. *IEEE Network*, Vol. 32, No. 1, pp. 96–101, 2018.

[24] Bo Tang, Zhen Chen, Gerald Hefferman, Tao Wei, Haibo He, and Qing Yang. A hierarchical distributed fog computing architecture for big data analysis in smart cities. In *Proceedings of the ASE BigData & SocialInformatics 2015*, p. 28. ACM, 2015.

[25] Shusen Yang. Iot stream processing and analytics in the fog. *IEEE Communications Magazine*, Vol. 55, No. 8, pp. 21–27, 2017.

[26] S. Chintapalli, D. Dagit, B. Evans, R. Farivar, T. Graves, M. Holderbaugh, Z. Liu, K. Nusbaum, K. Patil, B. J. Peng, and P. Poulosky. Benchmarking streaming computation engines: Storm, flink and spark streaming. In *2016 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pp. 1789–1792, 2016.

[27] J. Karimov, T. Rabl, A. Katsifodimos, R. Samarev, H. Heiskanen, and V. Markl. Benchmarking distributed stream data processing systems. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, pp. 1507–1518, 2018.