Vol.2019-GN-106 No.28
Vol.2019-CDS-24 No.28
Vol.2019-DCC-21 No.28
2019/1/25

フォグコンピューティング向け水平統合型 IoT プラットフォームの提案とその評価

板垣弦矢[†1]　撫中達司[†1]　齊藤志保[†2]　折本拓真[†2]　森郁海[†2]　伊藤岳広[†2]

概要：近年，エンドデバイスに近い端末だけでサービスにおける必要最低限の機能を提供することを目的としたフォグコンピューティングと呼ばれるアーキテクチャが注目されている．フォグコンピューティングではフォグノードと呼ばれる端末が端末間のスペックの差などを吸収しながら分散的に接続し様々なドメインのタスクを共有しあうため，システムを構築する上では共通した機能をまとめる水平性が重要であるとされている．しかし，クラウドコンピューティング上でサービスを提供するにあたり機械学習を用いることは一般的であるがフォグコンピューティングに機械学習を実装した例はあまりない．これは従来の機械学習が特定の目的を果たすために専門的に学習を行ういわゆる垂直型であることが原因と推察できる．システムに水平性をもたせる方法としてはオントロジーを用いることが推奨されており，その一例として標準化団体である oneM2M が提案する水平統合型 IoT プラットフォームがある．これは様々なドメインが共通する部分に機械学習を用い，特定ドメインへの拡張についてはオントロジーを用いることで水平性を実現したプラットフォームであるが，フォグコンピューティング上の機械学習に水平性をもたせる提案としては詳細な設計がされておらず，開発者に委ねられている部分が大きいため課題が残っている．そこで我々は当該プラットフォームをフォグコンピューティングへ適用した際に生じる考慮すべき点を明らかにし，フォグノードのハードウェアスペックや求められる精度，拡張度，応答速度などのユースケースに応じた機械学習及びオントロジーの適切な調整について提案することを課題として，今回は医療ドメインをユースケースにフォグコンピューティング向けの水平性統合型 IoT プラットフォームを実装し，仮説検証のための計測，またその評価を行った．結果，考察通りプラットフォームの負荷を変化させることができた一方で考察に反した結果も得られたことから機械学習及びオントロジーの適切な調整について提案することはできなかった．使用するデータやライブラリの違いがどのように影響するか調査することが今後の課題である．

キーワード：フォグコンピューティング，機械学習，オントロジー，水平統合型 IoT プラットフォーム

1. はじめに

近年，フォグコンピューティングと呼ばれるアーキテクチャがクラウドコンピューティングにおける応答性や可用性，プライバシーなどの問題を解決するとして注目され[1], openFog といった国際標準化団体による標準化が進められている[2]．フォグコンピューティングに似たアーキテクチャの 1 つにエッジコンピューティングと呼ばれるアーキテクチャが存在するが，これはフィルタリングや暗号化など比較的軽い処理をエッジ端末に任せ，重い処理をクラウドに任せるいわゆる作業分担を行うことで負荷分散を主に狙ったアーキテクチャであったのに対し，フォグコンピューティングではエッジ端末をフォグノードと呼び，それらを分散的に接続しタスクを共有し合うことでフォグノードだけでサービスにおける必要最低限の機能を提供することを目的としている．フォグノードにはルーターやゲートウェイ，PC，スマートフォンなど様々な端末が用いられることを想定しており，そのため異なるスペックの端末間でもタスクの共有が行えるようシステムを構築する上では共通する機能をまとめる水平性が重要であるとされている[3].

しかし，クラウドコンピューティング上でサービスを提供するにあたり機械学習を用いることが一般的である一方，機械学習をフォグコンピューティングに実装した例は多くない．これは従来の機械学習が特定の目的を果たすために専門的に学習を行ういわゆる垂直型であることが原因であると推察できる．システムに水平性をもたせる方法としてはオントロジーを用いることが推奨されており，その一例として標準化団体である oneM2M では機械学習とオントロジーを組み合わせた水平統合型 IoT プラットフォームを提案している[4][5]. このプラットフォームにおいて機械学習は様々なドメインが共通して扱う事物を予測するものとしての役割を担っており，その機械学習の出力をもとにオントロジーを用いて特定のドメインへの拡張を行っている．しかし，当該プラットフォームはフォグコンピューティング上の機械学習に水平性をもたせる提案としては詳細な設計がされておらず，開発者に委ねられている部分が大きいため課題が残っている．

そこで本稿では当該プラットフォームをフォグコンピューティングへ適用した際に生じる考慮すべき点を明らかにし，フォグノードのハードウェアスペックや求められる精度，拡張度，応答速度などのユースケースに応じた機械学習及びオントロジーの適切な調整について提案することを課題として，今回は医療ドメインをユースケースに設計したフォグコンピューティング向けの水平性統合型 IoT プラットフォームにおいての計測結果及び評価結果を示す．

†1 東海大学 情報通信学研究科
Graduate School of Information and Telecommunication Engineering, Tokai University
†2 三菱電機株式会社 情報技術総合研究所
Mitsubishi Electric Corporation Information Technology R&D Center

Vol.2019-GN-106 No.28
Vol.2019-CDS-24 No.28
Vol.2019-DCC-21 No.28
2019/1/25

2. 水平統合型 IoT プラットフォームをフォグへ適用するための提案と考慮すべき点について

oneM2M の提案する水平統合型 IoT プラットフォームはフォグノードにおけるハードウェアの性能まで考慮した詳細な設計がされておらず，前述した通りフォグノードには様々な種類の端末が用いられることから特にハードウェアの性能が低いものには適用することが難しいと考えられる．一方で当該プラットフォームは主に機械学習とオントロジーから構成されるため，図 1 の様にこれらのうちのどちらかもしくは両方のハードウェアに対する負荷をユースケースに合わせて調整することができればフォグコンピューティングに対して当該プラットフォームを適用する事ができる可能性がある．

そこで本稿では機械学習及びオントロジーの負荷調整による水平統合型 IoT プラットフォームのフォグコンピューティング適用を提案し，それぞれがハードウェアに対して与える影響を考慮すべき点として捉え考察を行う．なお，ここでのハードウェアとはコンピュータにおける 5 大装置のうちの制御装置，演算装置，記憶装置を引用しており，制御装置，演算装置を CPU，記憶装置をメインメモリとストレージとして解釈している．

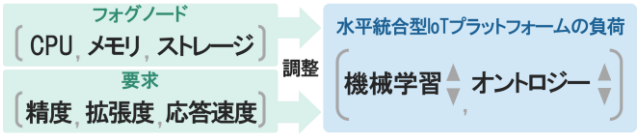

図 1　水平統合型 IoT プラットフォームの負荷調整

2.1 機械学習について

機械学習は学習アルゴリズムによって推論の方法や推論に必要となるデータの保持の仕方が異なることから CPU やメモリ，ストレージなどハードウェアに対して与える負荷が総合的に変化することが考えられる．oneM2M の提案プラットフォームではオントロジーは機械学習から特定のワードを受け取る設計となっていることから教師あり学習の分類アルゴリズムを対象として表 1 のようにカテゴライズした．

表 1　負荷を観点とした分類アルゴリズムのカテゴライズ

	分類アルゴリズム		
負荷	線形	非線形	
	小	線形ベース	その他
		中	中

分類には大きく線形分類と非線形分類が存在し，線形分類に関しては 1 次関数で表現することができるような単純なアルゴリズムであることから比較的負荷が小さいと考えられる．また，その応用としてカーネル関数や多項式を用いて線形分離不可能な問題に対して分類を行う線形ベースの非線形の分類アルゴリズムについては応用を加えたアルゴリズムであるかそうでないかとういう関係からハードウェアに与える負荷としては，線形ベース＞線形のような関係が成り立つと考えられる．その他にも非線形の分類アルゴリズムについては近傍や分類木を使う様々なアプローチが存在し，いずれもデフォルトで線形分離不可能な問題に対応できるアルゴリズムであることから少なくとも線形のアルゴリズムより負荷が大きいことが考えられる．

以上のことから分類アルゴリズムがハードウェアに与える負荷を観点とすると表 1 のようにカテゴライズすることができるが，機械学習を用いる上では他にも要求される精度や応答速度があり，例えば前述した線形の分類アルゴリズムについてはハードウェアに対する負荷が 1 番小さいことが想定される一方で線形分離不可能な問題に対しては高い精度を出すことができないことが考えられる．このことから機械学習においては扱うデータと要求される精度や応答速度を考慮しつつフォグノードのハードウェアスペックに合わせたアルゴリズムを選択する必要があると考えられる．

2.2 オントロジーについて

オントロジーはデータとデータの関係性を記述することで任意のドメイン知識を表現しているという特性上，ドメインの対象とする知識範囲が広い場合にはそれに応じてオントロジーのファイルサイズが変化すると考えられる．一般的なオントロジーの使い方ではストレージにオントロジーのファイルが収まるかどうかだけが重要であり負荷調整を行うことはできないため，フォグノードのストレージに合わせてオントロジーのファイルサイズを調節することを想定した場合，大きなオントロジーからより必要となる知識だけを切り出す技術が必要であると考えられる．他にもオントロジーのクエリにおいては最終的に得られる結果の規模によって CPU やメモリに与える負荷が変化することが考えられ，これに対しオントロジーのクエリ言語として一般的に使用される SPARQL では得られる結果の規模を制限する事が可能であるが，出力される結果が元のワードからどの程度拡張されたかや，ワードがどのような関係をたどって拡張されたかを確認することができない．水平統合型 IoT プラットフォームにおけるオントロジーの役割がワードの拡張であることを考慮すると結果の制限と同等にこれらの機能は重要であると考えられる．オントロジーの入力に使う元のワードから繋がりのあるワードを見つけ，結果を広げていくその程度のことを拡張度として表した場合，拡張度が上がるに連れて得られる結果の規模が大きくなることが予想される．このことからオントロジーにおいては拡張度をパラメータとしてクエリを行う技術を用い，要求される拡張度や応答速度，フォグノードのハードウェアス

Vol.2019-GN-106 No.28
Vol.2019-CDS-24 No.28
Vol.2019-DCC-21 No.28
2019/1/25

ペックなどに応じて拡張度の調整を行う必要があると考えられる．

3. ユースケース

2 章での考慮点を検証するにあたり，本章では検証のためのユースケースを設定する．

前述した oneM2M や openFog はそれぞれユースケースの 1 つとして医療分野を取り上げている．その理由として近年，全世界的に予防型医療へのパラダイムシフトが望まれているという背景がある[6]．これまでの医療システムは基本的に診断と治療をメインに行うものであり，予防に関しては力を入れていなかった．しかし，近年，生活習慣病といった本来予防可能な慢性疾患の患者数が増加し，医療費の増大や死亡者数の増加を招いていることから慢性疾患の患者数減少を目的として予防型医療が注目されており，効率的な予防型医療の実現手段として医療と IT 技術をかけ合わせた e ヘルスケアに関する研究が盛んに行われている[7][8]. 中でもフォグコンピューティングにおいては患者のバイタルをリアルタイムに取得できるという特徴から異常があった際に迅速に対応することが可能であり且つ可用性が失われにくいという点が高く評価され，多くの研究で適用の必要性が語られている[9][10].

しかし，e ヘルスケアの中でも疾病予測と呼ばれるようなタスクにおいては現状，クラウドに貯めた大量の電子カルテを用いて実現されることが多い．高齢化が進むに連れて疾病予測の需要が高まった際には，クラウドコンピューティング型のアーキテクチャでは負荷集中による応答性や可用性の低下が考えられ，こと電子カルテにおいては既往歴などのプライベートな情報が載っていることからプライバシー保護に関する配慮も必要であると考えられる．

以上のことから本稿においては，主に生活習慣病をターゲットとした疾病予測をユースケースとして設定する．

また，フォグノードの選定においては前提条件としてユースケースのターゲットユーザーが家庭であることから評価項目を価格，大きさ，消費電力とし，結果，各項目においてバランスが取れ，かつ一般的に使用されている Raspberry Pi 3 をフォグノードとして設定した．搭載する OS についてはラズベリーパイ財団公式 OS である Raspbian を使用し，ストレージは 4GB 以上が推奨されていることから 8GB のものを使用することとした．

4. 実装

前述したクラウドコンピューティングの疾病予測では電子カルテに記載されたバイタルデータを用いているが，生活習慣病の患者が増加し予防型医療へのパラダイムシフトが望まれている背景として根本的に病気に対する意識が低く病院へ足を運ばないケースが多いことが推察される．また，ウェアラブルセンサーなどを用いた病院外でのバイタルデータ取得についても意識の変化がない限り積極的に身につける可能性は低いと考えられる．以上のことから予防型医療を目指した疾病予測においては普段の生活に溶け込ませる形でデータを取得する必要があると考えられる．そこで本実装においては生活行動を間接センシングと機械学習により取得した後，疾病に関する知識が記述されたオントロジー用いることによって発症する恐れのある疾病を予測する疾病予測システムを実装する．

4.1 機械学習

今回設定した疾病予測システムにおける機械学習の役割は間接センシングにより得たセンサデータを用いて生活行動を予測することである．そのため本来であればどのようなセンサデータを用いるのか，どのような特徴を抽出するのかなどについて議論する必要がある．しかし，本論文のスコープとするところは水平統合型 IoT プラットフォームであり疾病予測のためのセンサデータ取得ではない．このことから今回はセンサデータの取得部分には言及しないこととし，インターネット上に公開されているすでに特徴量抽出の完了した行動認識用のデータセットを代替データとして利用した[11]．また，各学習アルゴリズムのパラメータ調整につてはスコープ内ではあるものの，まずは 2.1 節で考察したカテゴライズについての検証を優先するべきと考え，本実装においてはすべてデフォルトで設定した．

4.2 オントロジーの切り出し

2.2 節の考察ではストレージに対する負荷調整の技術としてオントロジーの切り出しの必要性を提案した．本実装ではこのアイデアをもとに医療に関する大きなオントロジーから疾病部分だけを切り出すことを想定して図 2 に示す疾病オントロジーを自作した．

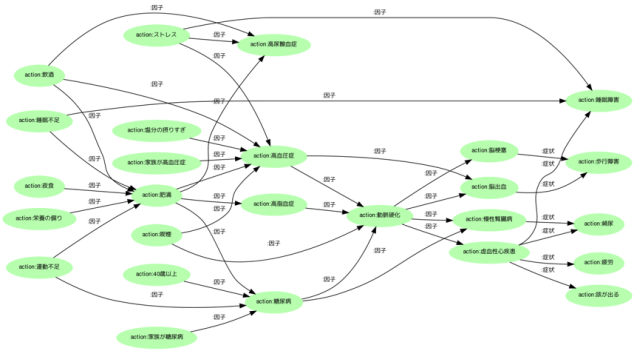

図 2　自作した疾病オントロジー

4.3 オントロジーのクエリにおける再帰的処理

また 2.2 節では，拡張度をパラメータとしたクエリについて言及したが，今回このアイデアについては再帰的処理を用いることで実装を行った．入力されたワードと順方向に繋がりのあるワードをすべてクエリすることを基本機能として，拡張度が 2 以上であった場合に出力された結果に対してもう一度同じ処理を行う再帰的な処理を行うことで

Vol.2019-GN-106 No.28
Vol.2019-CDS-24 No.28
Vol.2019-DCC-21 No.28
2019/1/25

図 3 のように得られる結果の調整を実現している．またこれによりどのような関係をたどってワードが拡張されているのかを示すことが可能となり，特定のワードの繋がりについては除外するといったようなフィルタリングを行うことも可能となった．

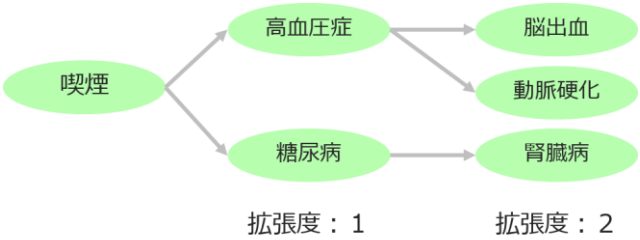

図 3　拡張度をパラメータとしたクエリ

しかし本実装においては拡張度が上がるに連れて本来の目的とは関係のないワードが含まれやすくなってしまう．そこで，我々はこの問題に対応するためクエリにおける重み付け処理の実装も行った．この重み付け処理では再帰的処理の中で同じワードが出現した回数に着目しており，例えば図 4 では拡張度を 1 として運動不足のクエリを行った場合，肥満と糖尿病の重みは 1 と出力される．しかし，拡張度を 2 としてクエリを行った場合には，新たに肥満のクエリが行われ糖尿病の出現回数が 2 回となり結果，重みが 2 となる．この様に重み付け処理を用いることで元のワードと出力されるワードとの関係性が数値として明確になり重みの低いものについては除外するといった処理を行うことが可能である．

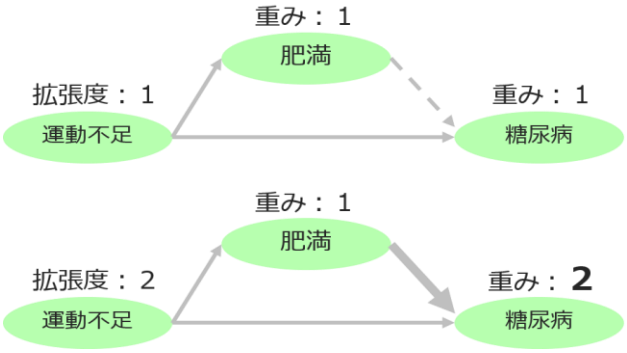

図 4　クエリにおける重み付け処理

5. 計測

5.1 計測目的

2 章では大きく機械学習とオントロジーについて言及した．機械学習については学習アルゴリズムの違いによってハードウェアに対する負荷が変わることを考察したことから本計測ではサーベイ論文[12]で取り上げられている学習アルゴリズムを元に表 2 に示す 8 つの学習アルゴリズムで計測を行うこととし，フォグノードのハードウェアとして設定した CPU，メモリ，ストレージに関連する処理時間，メモリフットプリント，必要記憶容量の違いについて確認することを計測の目的とした．

表 2　計測する学習アルゴリズム

線形		線形 SVM
		ロジスティック回帰
		ナイーブベイズ
非線形	線形ベース	SVM（RBF カーネル）
		多項ロジスティック回帰
	その他	k 近傍法
		ランダムフォレスト
		ニューラルネットワーク

オントロジーについては切り出しと拡張度の調整によってハードウェアに対する負荷が変わると考察していたが，今回は切り出しについて技術的な実装を行っていないことからオントロジーのファイルサイズについては 1 パターンのみ計測することとし，拡張度については拡張度 1 でクエリを行った場合と自作した疾病オントロジーをすべて探索することのできる拡張度 5 でクエリを行った場合の 2 パターンで計測を行い，処理時間，メモリフットプリント，必要記憶容量の変化について確認することを計測の目的とした．

また，本稿の最終課題であるユースケースに応じた機械学習及びオントロジーの適切な調整についての提案の知見とするため，実装したプラットフォームが設定したユースケースの求める応答速度を満たすことができるのか，フォグノードとして設定した Raspberry Pi 3 のメモリ及びストレージに収まるのかについても計測の目的とした．具体的な上限値を表 3 に示す．

表 3　ユースケースにおける計測対象の上限値

計測対象	上限値
処理時間	180 [s]
メモリフットプリント	710 [MB]
必要記憶容量	3 [GB]

応答速度はより厳しい条件下で評価をするべきという考えのもと疾病予測を拡張した症状予測を行いアラートのような迅速な応答が可能かを評価するため消防庁も引用しているカーラーの救命曲線[13]から生存率 50%のラインである 180 秒を上限値として設定した．メモリは Raspberry Pi 3 のメモリ容量から OS 分を差し引いた 710MB を上限値に設定し，ストレージについてもストレージ容量から OS 分を差し引いた 3GB を上限値に設定した．

Vol.2019-GN-106 No.28
Vol.2019-CDS-24 No.28
Vol.2019-DCC-21 No.28
2019/1/25

5.2 計測環境

計測対象の疾病予測システムを python3.5 でフォグノードである Raspberry Pi 3 上に実装する．以下，疾病予測システムの処理の流れ及び使用したライブラリを図 5，表 4 に示す．

図 5 疾病予測システムにおける処理の流れ

ライブラリ名	バージョン	用途
scikit learn	0.19.2	機械学習
scipy	1.1.0	依存[scikit learn]
numpy	1.15.0	依存[scikit learn]
pickle	4.0	モデルのバイナリ化
rdflib	4.2.2	オントロジー
pyparsing	2.2.0	依存[rdflib]
isodate	0.6.0	依存[rdflib]
six	1.11.0	依存[rdflib]

表 4 疾病予測システム実装に使用したライブラリ

各計測項目の計測の方法については基本的に計測ツールを用いて機械学習及びオントロジーの処理に関わるプログラムを一行ずつ計測することとし，メモリフットプリントは memory-profiler(0.54.0)，処理時間は line-profiler(2.1.2) を使用することとした．必要記憶容量については直接ファイルサイズを確認することとし，モデルのファイルサイズやオントロジーのファイルサイズに加えて上記ライブラリも含め計測を行うこととした．

6. 結果

6.1 機械学習部分についての計測結果

各学習アルゴリズムの計測結果を表 5 から表 12 に示す．図 6 に示す通りモデルのファイルサイズについては概ね考察通り線形と非線形のアルゴリズムで差が出ており非線形のなかでもその他に分類されるアルゴリズムについては線形との差が顕著に現れる結果となった．しかし線形ベースについては SVM（RBF カーネル）と多項ロジスティック回帰で大きくファイルサイズが異なる結果となった．処理時間についてモデルのロードでは図 7 に示す通り考察に反する結果となったが一方で予測では概ねファイルサイズに対応する形で処理時間が変化した．メモリフットプリントについてモデルのロード時では図 8 に示す通り SVM（RBF カーネル），k 近傍法を除いて大きく差が生じることはなく考察と異なる結果となった．また，予測部分では SVM（RBF カーネル）を除いて負荷を計測することができなかった．

表 5 線形 SVM の計測結果

線形 SVM		
精度	0.78 (+/- 0.10)	
モデルのファイルサイズ	28[KB]	
	モデルのロード	予測
処理時間	3.540[s]	0.001[s]
メモリフットプリント	23.211[MB]	0.000[MB]

表 6 ロジスティック回帰の計測結果

ロジスティック回帰		
精度	0.79 (+/- 0.10)	
モデルのファイルサイズ	28[KB]	
	モデルのロード	予測
処理時間	3.511[s]	0.001[s]
メモリフットプリント	23.406[MB]	0.000[MB]

表 7 ナイーブベイズの計測結果

ナイーブベイズ		
精度	0.52 (+/- 0.22)	
モデルのファイルサイズ	54[KB]	
	モデルのロード	予測
処理時間	3.198[s]	0.004[s]
メモリフットプリント	18.852[MB]	0.000[MB]

表 8 SVM（RBF カーネル）の計測結果

SVM（RBF カーネル）		
精度	0.76 (+/- 0.15)	
モデルのファイルサイズ	14[MB]	
	モデルのロード	予測
処理時間	3.548[s]	0.017[s]
メモリフットプリント	37.113[MB]	0.254[MB]

表 9 多項ロジスティック回帰の計測結果

多項ロジスティック回帰	
精度	0.82 (+/- 0.11)
モデルのファイルサイズ	28[KB]

Vol.2019-GN-106 No.28
Vol.2019-CDS-24 No.28
Vol.2019-DCC-21 No.28
2019/1/25

	モデルのロード	予測
処理時間	3.526[s]	0.001[s]
メモリフットプリント	23.129[MB]	0.000[MB]

表 10　k 近傍法の計測結果

k 近傍法		
精度	0.73 (+/- 0.07)	
モデルのファイルサイズ	39[MB]	
	モデルのロード	予測
処理時間	3.460[s]	0.039[s]
メモリフットプリント	59.770[MB]	0.000[MB]

表 11　ランダムフォレストの計測結果

ランダムフォレスト		
精度	0.74 (+/- 0.13)	
モデルのファイルサイズ	651[KB]	
	モデルのロード	予測
処理時間	3.438[s]	0.010[s]
メモリフットプリント	23.773[MB]	0.000[MB]

表 12　ニューラルネットワークの計測結果

ニューラルネットワーク		
精度	0.77 (+/- 0.11)	
モデルのファイルサイズ	1.4[MB]	
	モデルのロード	予測
処理時間	3.334[s]	0.003[s]
メモリフットプリント	21.977[MB]	0.000[MB]

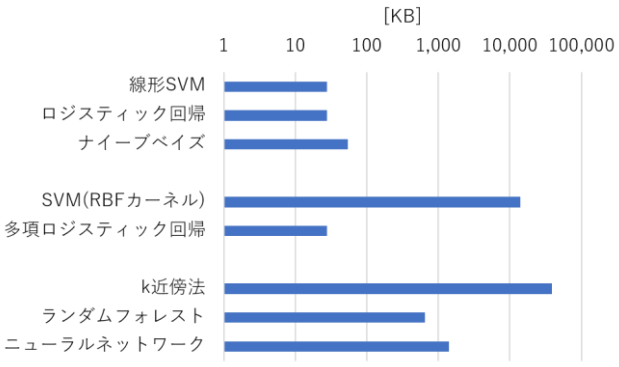

図 6 各学習アルゴリズムのモデルのファイルサイズ

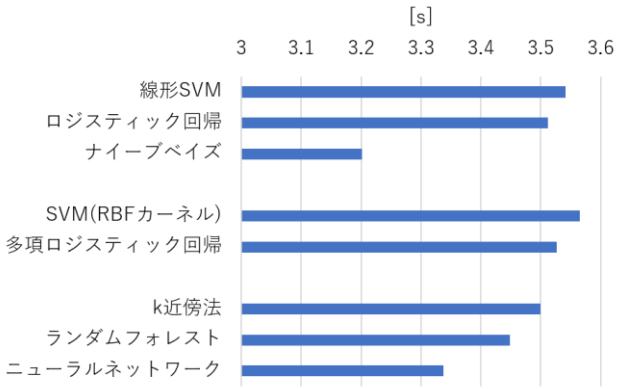

図 7 各学習アルゴリズムの合計処理時間

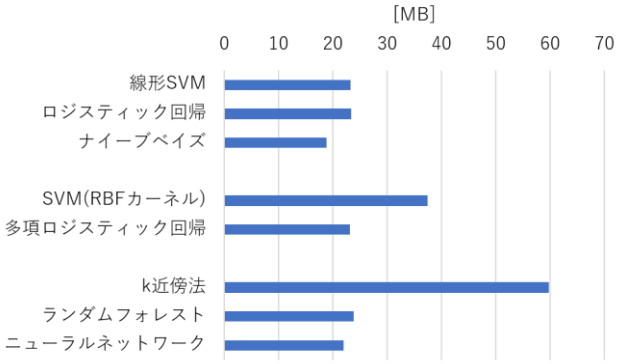

図 8 各学習アルゴリズムの合計メモリフットプリント

6.2 オントロジー部分についての計測結果

オントロジー部分についての計測結果を表 13 に示す. 図 2 に示した自作した疾病オントロジーではファイルサイズが 3[KB]と非常に軽量であった．また，処理時間に関しては考察通り拡張度が増すと，それに伴い計測結果に変化が出る結果となった．しかしメモリフットプリントについては考察に反し，ともに 0.000[MB]となったため負荷を計測することができなかった．

表 13　オントロジーの計測結果

オントロジーのファイルサイズ	3[KB]	
	再帰的処理	
	拡張度 1	拡張度 5
処理時間	0.001[s]	0.009[s]
メモリフットプリント	0.000[MB]	0.000[MB]

6.3 ユースケースについての計測結果

各学習アルゴリズムの違いや拡張度の違いについてはすでに述べているためここでは学習アルゴリズムに線形

Vol.2019-GN-106 No.28
Vol.2019-CDS-24 No.28
Vol.2019-DCC-21 No.28
2019/1/25

SVM を選択し拡張度を 5 に設定したプラットフォームを代表として実装したプラットフォーム全体の処理時間やメモリフットプリント，必要記憶容量について表 14 に示す．今回計測を行った学習アルゴリズムや拡張度などすべての組合せにおいて設定した上限値を超えることなくプラットフォームを動作させることができた．

表 14　実装したプラットフォーム全体の計測結果

線形 SVM　×　拡張度 5		
必要記憶容量 (上限値)	100[MB] (3GB)	
	処理時間 [s]	メモリフットプリント [MB]
ライブラリのロード	0.238	36.770
モデルのロード	3.548	23.730
予測	0.001	0.000
オントロジーのロード	0.236	0.168
再帰的処理	0.009	0.000
合計 (上限値)	4.032 (180)	60.668 (710)

また、図 9 に示す通り処理時間，メモリフットプリントともにライブラリとモデルのロードが負荷の主な原因となっており，特に処理時間においてはモデルのロードが支配的であった．

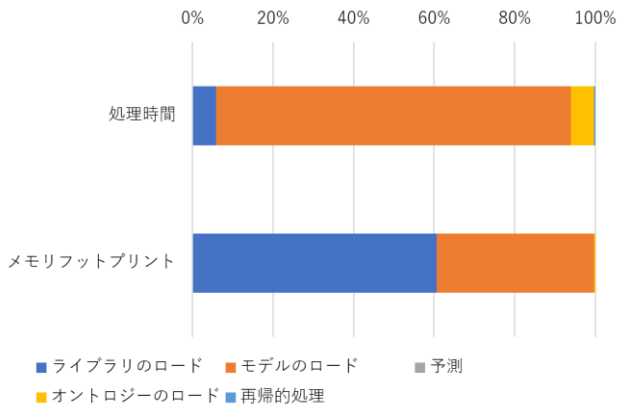

図 9 処理時間及びメモリフットプリントの内訳

7. 考察

機械学習のモデルのファイルサイズについては計測結果から概ね考察通りであったと考えられる．しかし，SVM（RBF カーネル）と多項ロジスティック回帰が考察に反し大きく差が出る結果となった．これについては特に SVM（RBF カーネル）はカーネル関数を用いて次元数を増やし線形分離不可能な問題に対応している事や，one vs rest で他クラス分類に対応していることが原因であると考えられ，データが密に詰まった，より線形分離の難しい分類に対してはカーネル関数を用いるとストレージを圧迫する可能性があると考えられる．多項ロジスティック回帰が線形のアルゴリズムと同じファイルサイズであったことについてはファイルサイズが 28[KB]と全く同じであることから機械学習のライブラリである scikit-learn もしくはモデルのバイナリ化に使用している pickle がファイルサイズを一定領域確保してモデルを作成していると考えられ，今回の実装におけるモデルの最小ファイルサイズは 28[KB]であったと考えられる．処理時間におけるモデルのロードやメモリフットプリントが考察と反していたことについても scikit-learn によるメモリの最適化などが影響した可能性がある． また，k 近傍法が他の学習アルゴリズムと比べ特別にメモリや記憶容量を消費していたことについては，アルゴリズムの特性上，全学習データの座標を保持していることが原因であると考えられる．学習のデータ量に応じてメモリやストレージに与える負荷が上昇する恐れがある学習アルゴリズムについてはフォグノードのようなハードウェアスペックの限られた端末には実装が難しく，このことからフォグコンピューティング向け水平統合型 IoT プラットフォームの機械学習において計測した学習アルゴリズムの中では k 近傍法を除いた 7 つの学習アルゴリズムが使用できると考えられる．

オントロジーの計測結果については処理時間が考察通りであったことから拡張度は CPU のスペックや求められる応答速度によって調整しなければいけないことが分かったが，一方でメモリフットプリントが考察に反する結果となったことについては最終的に得られた結果が小さすぎたために用いたツールでは計測が行えなかったことが原因と考えられる．

最後に実装したプラットフォーム全体の計測結果を通しては今回計測した処理時間のうちの大部分をライブラリやモデル，オントロジーのロードが占めていたことから，これについては予めロードを済ませておくことで大幅に処理時間を縮める事ができる可能性があり，より厳しい条件の要求にも対応することができるようになると考えられる．

8. まとめと今後の課題

本稿では水平統合型 IoT プラットフォームをフォグコンピューティングへ適用した際に生じる考慮すべき点を明らかにし，フォグノードのハードウェアスペックや求められる精度，拡張度，応答速度などのユースケースに応じた機械学習及びオントロジーの適切な調整について提案することを課題として，設定したユースケースをもとに設計したフォグコンピューティング向けの水平性統合型 IoT プラットフォームの計測評価を行った．結果，2 章での考察通りハードウェアの負荷を調整させることができた項目があった一方で考察に反した項目もあったことからユースケースに応じた機械学習及びオントロジーの適切な調整を提案す

Vol.2019-GN-106 No.28
Vol.2019-CDS-24 No.28
Vol.2019-DCC-21 No.28
2019/1/25

ることはできなかった．水平統合型 IoT プラットフォームをフォグコンピューティングへ適用した際に生じる考慮すべき点をより明らかにするため，使用するデータやライブラリの違いがどのようにハードウェアに対して影響を与えるのか調査することが今後の課題である．

参考文献

[1] Ranesh Kumar Naha, Saurabh Garg, Dimitrios Georgakopoulos et al.. Fog Computing: Survey of Trends, Architectures, Requirements, and Research Directions. 2018.
[2] OpenFog Consortium. IEEE adopts OpenFog Reference Architecture as official standard for fog computing, Retrieved 12/17/2018, https://www.openfogconsortium.org/news/ieee-adopts-openfog-reference-architecture-as-official-standard-for-fog-computing/.
[3] Mohammad Saeid Mahdavinejad, Mohammadreza Rezvan, Mohammadamin Barekatain et al.. Machine learning for internet of things data analysis: a survey. Digital Communications and Networks. 2018, Volume 4, Issue 3, August, Pages 161-175
[4] oneM2M., oneM2M Ontologies, Retrieved 12/17/2018, http://www.onem2m.org/technical/onem2m-ontologies.
[5] 稲田修一，インプレス標準教科書シリーズ M2M/IoT 教科書，インプレス．
[6] 厚生労働省，予防・健康づくりの取組の推進, 2016, https://www5.cao.go.jp/keizai-shimon/kaigi/special/reform/wg1/280930/sankou1.pdf
[7] Diego Castro, William Coral, José Cabra et al.. Survey on IoT solutions applied to Healthcare. DYNA. 2017, 84(203), pp. 192-200
[8] LE Duy Tan.. A Survey on Internet of Things for Smart Health Technologies. JAIST Repository. 2018
[9] Behailu Negash, Tuan Nguyen Gia, Arman Anzanpour et al.. Leveraging Fog Computing for Healthcare IoT. Fog Computing in the Internet of Things. 2017, pp 145-169
[10] Frank Alexander Kraemer, Anders Eivind Braten, Nattachart Tamkittikhun et al.. Fog Computing in Healthcare—A Review and Discussion. IEEE Access. 2017, VOLUME 5, pp 9206-9222
[11] UCI.. Smartphone Dataset for Human Activity Recognition (HAR) in Ambient Assisted Living (AAL) Data Set, 2016, https://archive.ics.uci.edu/ml/datasets/Smartphone+Dataset+for+Human+Activity+Recognition+%28HAR%29+in+Ambient+Assisted+Living+%28AAL%29
[12] Carla Mouradian, Diala Naboulsi, Sami Yangui et al.. A Comprehensive Survey on Fog Computing: State-of-the-Art and Research Challenges. IEEE Communications Surveys & Tutorials. 2017, Volume: 20 , Issue: 1, pp 416-464
[13] 東京消防庁，救急医療週間の実施について, 2010, http://www.tfd.metro.tokyo.jp/hp-kouhouka/pdf/220902.pdf