データセット細分化を用いた 時系列データ回帰モデル化手法の検討

高橋 佑里子[1] 鈴木 成人[2] 山本 拓司[2] 福田 裕幸[2] 小口 正人[1]

概要：近年のクラウドサービスにおいて，サーバを仮想化することで使用率を向上させ，サーバ数を削減する取り組みが行われている．この取り組みでは，サーバが自身の CPU 資源を超えた CPU を割り当てられるオーバーコミット状態に陥ることで，仮想サーバの性能が低下する可能性があるため，制御対象のすべての仮想サーバの CPU 使用率を予測し制御を行う必要がある．本研究では，仮想サーバの CPU 使用率の汎用的な深層学習予測モデルの生成に向けて，時系列データの回帰モデル化手法についての検討を行う．方法を模索した結果，時系列データを学習に必要な長さごとに抽出を行い，細分化した後のデータをランダムに使用することで，再学習時に使用するデータ数の削減が可能であることを確認した．

Consideration of Time Series Regression Modeling Method Using Data Set Subdivision

YURIKO TAKAHASHI[1] SHIGETO SUZUKI[2] TAKUJI YAMAMOTO[2] HIROYUKI FUKUDA[2]
MASATO OGUCHI[1]

1. はじめに

近年のクラウドサービスにおいて物理サーバ (Physical Machine: PM) の CPU 使用率は低く，そのパフォーマンスを十分に発揮できない状態が続いている．これを改善すべく，事業者では，サーバを仮想化することで使用率を向上させ，PM 数を削減する取り組みが行われている．この取り組みでは，PM が自身の CPU 資源を超えた CPU を割り当てられるオーバコミット状態に陥ることで，仮想サーバ (Virtual Machine: VM) の性能が低下する可能性がある．そのため，図 1 のように制御対象のすべての VM の CPU 使用率を予測し，値が上昇する前に VM を別の PM へマイグレーションする等の制御を行う必要がある [1]．しかし，現状の CPU 使用率の予測モデルは汎用性が低く，特定の VM に対しては高い精度で予測できる一方，その他の VM に対する予測精度は低くなるという課題がある [2]．

本研究では，VM の CPU 使用率の汎用的な深層学習予測モデルの生成に向けて，時系列データの回帰モデル化手法の検討を行う．深層学習モデルは学習時間が長いため，少量の再学習での精度向上が求められる．そこで，データを適切に選定することで，再学習で使用するデータ数を削減する方法を検討した．その結果，時系列データを学習に必要な長さごとに抽出を行い，細分化した後のデータをラ

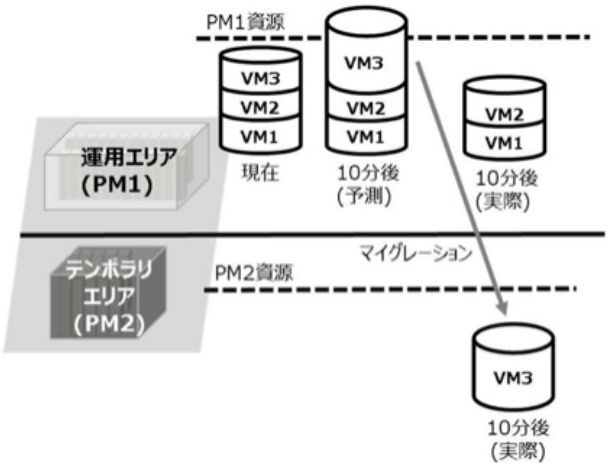

図 1 VM 制御のイメージ

[1] お茶の水女子大学
Ochanomizu University
[2] 株式会社富士通研究所
FUJITSU LABOLATORIES LTD.

ンダムに使用することで，再学習時に使用するデータ数の削減が可能であることを確認した．

2. 関連研究

クラウドサービスにおける VM の CPU 使用率の予測や制御に関する研究は，以前から多く行われている．

[3] では，実際のデータセンターから取得したデータトレースを使用し，VM のワークロードの特徴付けを行い，隠れマルコフモデルをベースとした手法でワークロードパターンの変動を予測する機能を開発している．[4] では，サービスレベル契約 (SLA) 違反と電力コストの削減を目的として，線形回帰手法に基づいて PM の CPU 使用率の短期的な予測を行っている．PM が過負荷になった場合，一部の VM を他の PM に移行することで，エネルギー消費量と SLA 違反率の大幅な削減に成功している．[5] では，VM の CPU 使用率などののバースト性があり，中には明確な周期性を持たない時系列データに対しても，将来の負荷やピーク負荷，タイミングを正確に予測できるニューラルネットワーククベースのフレームワークの開発を行っている．このフレームワークにより，ARIMA モデルや基本的なニューラルネットワークモデルと比較して予測誤差は最大 3 倍，予測タイミングは 2〜9 倍の改善に成功している．[6] では，エネルギー消費量に基づいて，リアルタイムに VM を統合するフレームワークの提案を行っている．提案されたフレームワークを使用することで，フレームワークを使用していないデータセンターと比較して最大 80%の改善が示され，PM の高使用率と省エネルギーの実現に成功している．

3. 関連技術

3.1 Recurrent Neural Network (RNN)

RNN とは，再帰型構造を持つニューラルネットワークであり，過去の計算情報を保持することができるため，時系列データの学習に用いられている．しかし，長期間の情報を保持できないという欠点がある．そこで，セルの内部に入力ゲート，出力ゲート，忘却ゲートを導入することで，RNN を長期依存を扱うことができるように改良したものが Long Short-Term Memory (LSTM) である．そして，LSTM の忘却ゲートと入力ゲートを単一の更新ゲートとして一つに統合することで，計算量が比較的少なくなるよう改良したものが Gated Recurrent Unit (GRU) である．本研究では，計算量を軽微にする目的で GRU を使用し，GRU を 2 層繋げた深層学習ネットワークを構築した．GRU の構造を図 2 に示す．

3.2 TFLearn

TFLearn[7] とは，Tensorflow[8] の上に構築された透過的かつ互換性のある深層学習ライブラリである．これを使用することで簡潔なコードで深層学習モデルを記述することでき，実験を迅速に行うことが可能になる．本研究では，TFLearn を使用して深層学習を行った．

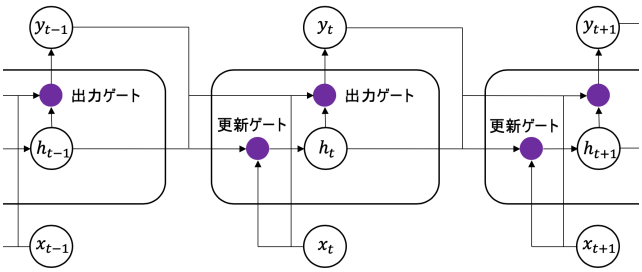

図 2　GRU の構造

3.3 fine tuning

fine tuning とは，転移学習の手法の 1 つである．ネットワークの上位層のみを一部を再学習することで，少量のデータの再学習で既存のモデルを別のターゲットに応用することが可能となる．

3.4 Root Mean Squared Error (RMSE)

RMSE は，二乗平均平方根誤差と呼ばれる回帰モデルの誤差を評価する指標の 1 つである．この数値が小さく 0 に近いほど，乖離が少なく精度が良いということになる．また，この数値は予測対象の波形の形状に依存する．長さ n の時系列データの正解値を y_t，予測値を $\tilde{y}_t$ とすると，以下のような式で計算される．

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(y_t - \tilde{y}_t)^2}$$

4. アプローチ

時系列データの学習を行う場合は通常，学習元データ数 (以下，l_h とする) と正解データ数を設定し，開始点を 1 点ずつずらしながら長さ l_h の学習元データとそれに対応する直後の長さ正解データ数の正解データのペアを作成し，学習に使用する．l_h を 10，正解データ数を 1 とした場合のイメージが図 3 である．この時系列データにおいて，グラフ下赤色の範囲が最初の学習元データ，グラフ中赤色の丸で囲ったデータが最初の正解データとなる．次の学習元データと正解データは 1 点ずらした黄色の部分，その次のデータはさらに 1 点ずらした緑色の部分，青色部分，紫色部分…となる．つまり，学習には長い波形をそのまま使うのではなく，細かい波形を多数使っているということになる．

そこで本研究では，この特徴を踏まえ，長い時系列データから細かい時系列データを事前に抽出し，細分化後にデータを選択することで，適切なデータ選定が可能になるのではないかと考えた．

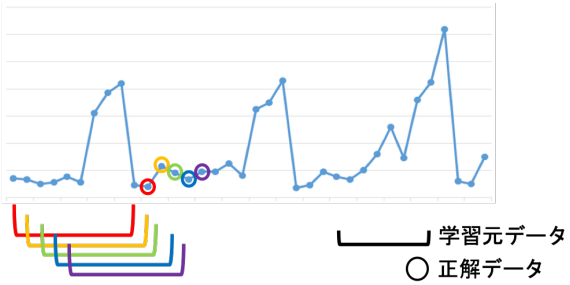

図 3　時系列データ学習のイメージ

5. 実験準備

5.1 実験に使用するデータ

本実験では，データセットとして Bitbrains IT Services Inc.[9][10] が公開しているデータセンタにおける VM 時系列データセットの ”CPU usage[%]” の項目を使用した．前処理として，時間間隔 300 ミリ秒，長さ約 43 分間のデータセットを，10 点ごとの平均値を取る平滑化処理と，最小値を 0，最大値を 1 に揃える正規化処理を行った．

その後，実験で使用するデータセットを選択するために，2000 個のデータセットに対し，クラスタ数を 20 として階層型クラスタリングを行った．デンドログラムは図 4 のようになった．縦軸はクラスタ間のユークリッド距離，括弧内の数字はそのクラスタに分類されたデータの個数である．この結果から，学習元データセット (以下 A とする) と A と似ているデータセット (以下 B とする) を赤色部分右側の山からそれぞれ 103 個，A と似ていないターゲットデータセット (以下 C とする) として赤色部分左側の山から 103 個のデータを選択した．A,B は同じ範囲からデータを取得しているが，データの重複はしていない．緑色部分のデータを使用しない理由は，全体的な動きが小さく，予測が容易であると考えられる波形が多く分類されていたためである．

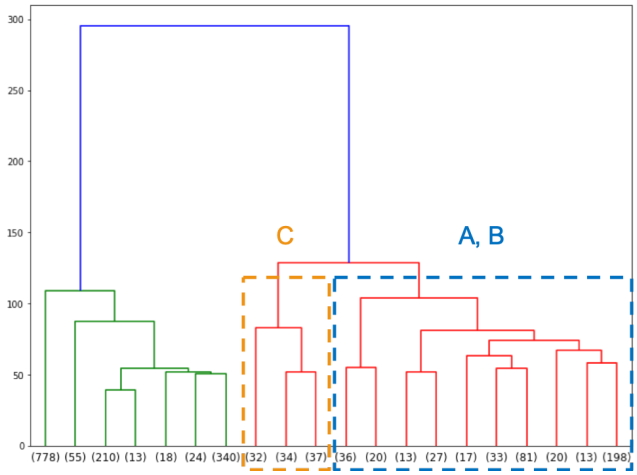

図 4　データセット全体の階層型クラスタリング結果と A,B,C の範囲

実際に使用したデータを以下に示す．図 5 が A の一例，図 6 が B の一例，図 7 が C の一例である．このように，A,B の概形は似ている一方，A,C の概形は似ておらず，A,B,C 全体の傾向としては，同じような挙動を繰り返すようなものが多数であった．

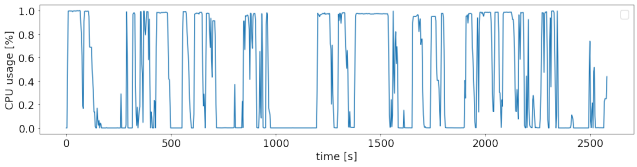

図 5　A の一例

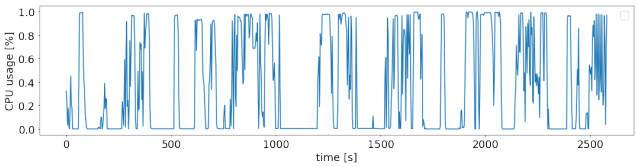

図 6　B の一例

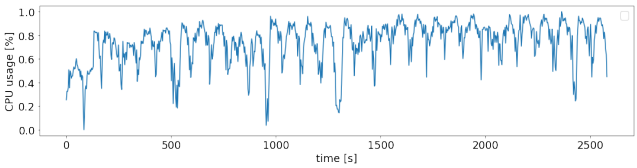

図 7　C の一例

5.2 データ粒度最適化

データを抽出する最適な長さを見つけるための l_h 比較実験を行った．代表として選んだ 10 個のデータのそれぞれで，正解データ数を 1 に固定し，前半 7 割を学習データとして l_h を変えて学習させ，後半 3 割をテストデータとして予測し，それらの結果 10 個の RMSE の平均値を比較した．実験結果は，図 8 のようになった．l_h が 24 のとき，RMSE の平均値が最も小さくなっている．

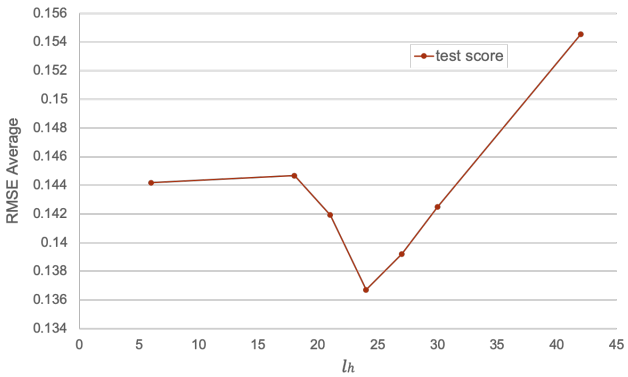

図 8　l_h 比較実験結果

この結果を受け，本実験の深層学習では，直前の 24 個のデータをもとに 1 個先のデータを予測するネットワークを構築することにした．また，データを抽出する長さは，学習元データ 24 点と正解データ 1 点を繋ぎ合わせた，合計 25 点とした．

5.3 データセット細分化

A,B,C に該当する合計 309 個のデータを，開始点を 1 点ずつずらしながら 25 点ごとに抽出し，細分化を行った．細分化後のデータセットの一部を図 9 に示す．同じ値の部分は同じ色で表している．

			0	1	2	3	4	...
cluster_name	data_number	position_number						
A	0	0	0.000056	0.000295	0.549699	0.996331	0.996857	...
		1	0.000295	0.549699	0.996331	0.996857	0.996881	...
		2	0.549699	0.996331	0.996857	0.996881	0.997759	...
		3	0.996331	0.996857	0.996881	0.997759	0.996195	...
		4	0.996857	0.996881	0.997759	0.996195	0.996929	...
...	...	...	...	...	...	...	...	...

図 9　細分化後のデータセットの一部

上記で抽出したデータを，k-means 法クラスタリングにより 30 個のクラスタに分類することで，3 種類のデータセット間の特徴を分析した．結果は図 10 のようになった．縦軸が分類された個数，横軸がクラスタ番号である．A と B は似ているデータセットのため同じような分布になっている一方，A と C は似ていないのデータセットのため異なる分布になっていることが分かる．

6. 実験

以下の 2 つの実験を行った．すべての実験において，学習および fine tuning を行う際のエポック数はいずれも 100 に統一しており，モデルの回帰精度の評価指標には，RMSE の平均値を用いている．

6.1 実験 1

実験 1 では，既存予測モデルは学習元データセットと似ているデータセットに対してどの程度の回帰精度を発揮するかを調査した．A で学習したモデルと B で学習したモデルのそれぞれで B を予測し，それらの回帰精度を比較した．

結果は表 1 のようになった．

表 1　実験 1 結果

学習元	予測対象	RMSE の平均値
A	B	0.1399
B	B	0.1381

6.2 実験 2

実験 2 では，既存予測モデルに対して fine tuning を行うことで別のターゲットに応用する適切な方法を検討した．A で事前学習したモデル (これを既存予測モデルとする) に C を使用した fine tuning を以下の 3 種類の方法で行い，その後，A で事前学習した fine tuning を行う前のモデル，方法 1,2,3 で fine tuning 行った後のモデル，および C そのもので学習したモデルのそれぞれで C を予測し，それらの回帰精度の比較を行った．

6.2.1 方法 1

方法 1 は，細分化後の C を k-means 法クラスタリングにより 30 個のクラスタに分類した結果 (図 11) を活用し，各クラスタ内から同一割合のデータを取得し，使用して fine tuning を行うというもので，5%，10%，15%，20%，40%，60%，80%の 7 パターンの実験を行った．データ取得前にクラスタ内でのデータのシャッフルは行っていない．各クラスタから取得したデータが均一に混ざるよう，データの順番のシャッフルを行い，各パターン試行回数 4 回の平均値を取得した．

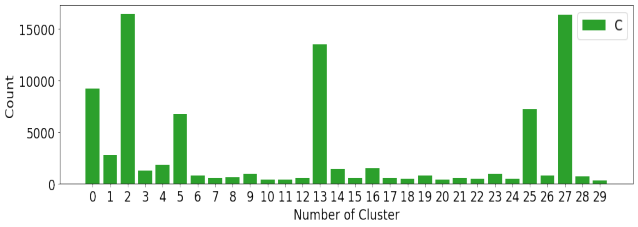

図 11　細分化後 C 内の k-means 法クラスタリング結果

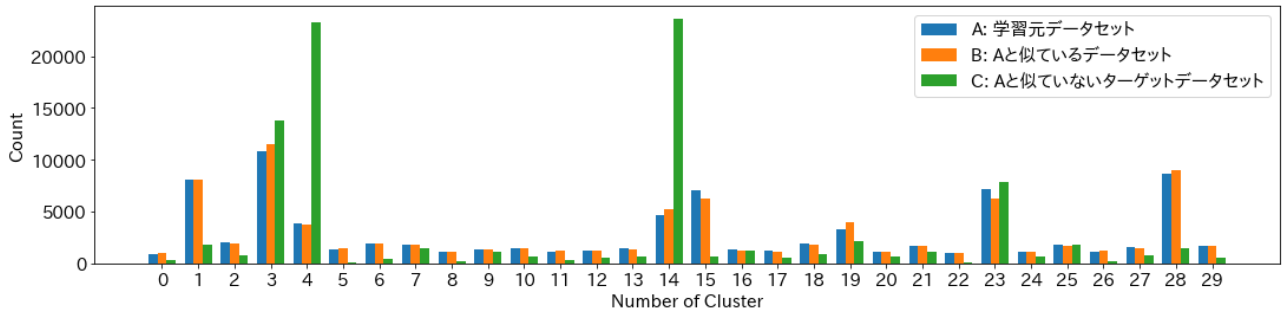

図 10　細分化後 A,B,C の k-means 法クラスタリング結果

6.2.2 方法 2

方法 2 は，細分化後のデータセットをランダムにシャッフル後，上から一定数のデータを使用して fine tuning を行うというもので，方法 1 と同様に 5%，10%，15%，20%，40%，60%，80%の 7 パターンの実験を行った．試行回数 4 回の平均値を取得した．

6.2.3 方法 3

方法 3 は，細分化を行わずに 103 個のデータからランダムにデータを選択，使用して fine tuning を行うというもので，方法 1,2 と同様に 5%，10%，15%，20%，40%，60%，80%の 7 パターンの実験を行った．試行回数 4 回の平均値を取得した．この方法では，細分化前の大きな単位でのデータ選択となるため，方法 1,2 の同一割合のパターンと学習に使用するデータ数は正確には一致していない．

6.2.4 結果

結果は図 12 のようになった．縦軸は RMSE の平均値，横軸は fine tuning 時に使用したデータの割合であり，左端の使用データ 0%の点が A で事前学習した fine tuning を行う前のモデルの結果，右端の使用データ 100%の点が C そのもので学習したモデルの結果である．

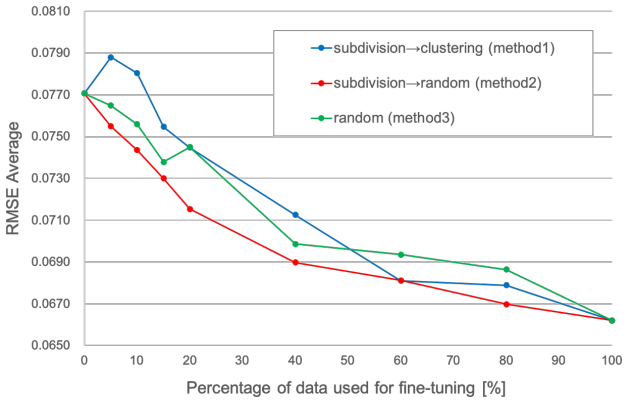

図 12　実験 2 結果

7. 考察

7.1 実験 1,2 の結果について

実験 1 の結果 (表 1) から，既存予測モデルは学習元データセットと似ているターゲットデータセットに対して，ターゲットデータセットで学習したモデルとほぼ同等の回帰精度を発揮することができるということが読み取れる．

実験 2 の結果 (図 12) において，各方法の同一割合パターンの結果を比較すると，細分化後のデータをランダムに使用する方法 2 のモデルが，細分化後のクラスタリング結果を活用する方法 1 のモデル，および細分化前の状態でランダムにデータを選定する方法 3 のモデルよりも精度が良くなることが読み取れる．また，0.069 の精度を達成するためには，方法 3 では約 80%のデータが必要だが，方法 2 では約 40%のデータで実現可能であることが読み取れる．このことから，データセットの細分化を行うことで，細分化を行わない通常のデータ選定よりも適切なデータ選定が可能になり，再学習時に使用するデータ数の削減が可能であることが分かる．

7.2 実験 2・方法 2 で使用したデータについて

実験 2 で，方法 2 の精度が最も良くなったことを受け，方法 2 で使用したデータのクラスタ番号分布の確認を行った．5%のパターンで使用された 4 種類のデータの分布を確認した結果が，図 13，図 14，図 15，図 16 である．これらの分布はいずれも，C 内でのクラスタリング結果である図 11 の分布と非常に似ており，5%以外のパターンについても同様の分布になっていることが判明した．各クラスタ内から同一割合のデータを取得する方法 1 よりも，ランダムにシャッフルする方法 2 の方が多種多様なデータが適切に混ざり，図 12 のような実験結果に繋がったと考えられる．

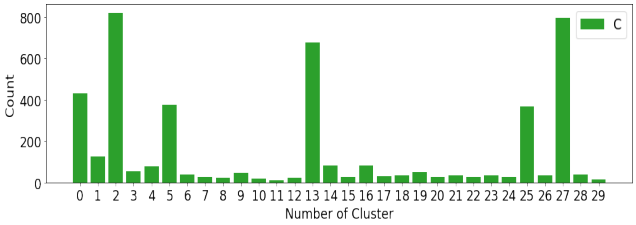

図 13　分布 1(5%)

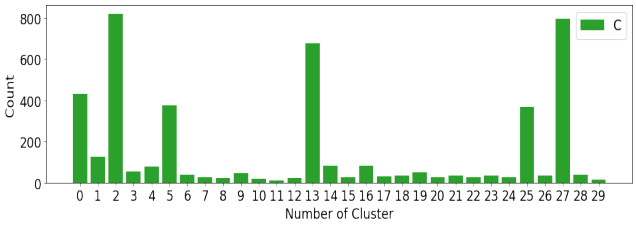

図 14　分布 2(5%)

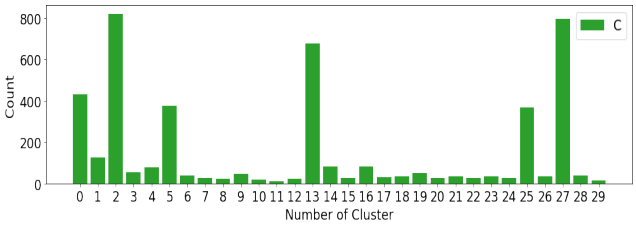

図 15　分布 3(5%)

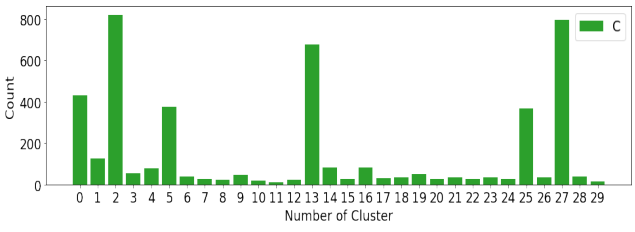

図 16 分布 4(5%)

8. まとめと今後の予定

VM の CPU 使用率の汎用的な深層学習予測モデルの生成に向けて，ターゲットデータセット内のデータを適切に選定し，再学習で使用するデータ数を削減する方法の検討を行った．実験の結果，似ているターゲットに対しては既存予測モデルをそのまま使用することができることを確認した．また，似ていないターゲットに対しては，学習に必要な長さごとに細分化を行った後のデータセットをランダムに使用して再学習を行うことで，細分化を行わない通常のデータ選定よりも適切なデータ選定が可能になり，再学習時に使用するデータ数の削減が可能であることを確認した．

今後は，予測モデルの長期運用を可能にする，コンセプトドリフト技術の検討を進めていきたいと考えている．

謝辞

本研究の一部はお茶の水女子大学と富士通研究所との共同研究契約に基づくものであり，JST CREST JPMJCR1503 の委託業務の助成を受けたものです．

参考文献

[1] 児玉宏喜, 鈴木成人, 福田裕幸, 吉田英司: マイグレーションを利用したデータセンタの高効率運用手法の提案とオーバコミット時における VM の性能評価, 情報処理学会論文誌 (2018)
[2] 鈴木成人, 児玉宏喜, 遠藤浩史, 福田裕幸: JIT モデリングによるサーバ負荷予測手法の検討と評価, 電子情報通信学会ソサイエティ大会 (2018)
[3] Khan, A., Yan, X., Tao, S., Anerousis, N. (2012, April). Workload characterization and prediction in the cloud: A multiple time series approach. In 2012 IEEE Network Operations and Management Symposium (pp. 1287-1294). IEEE.
[4] Farahnakian, F., Liljeberg, P., Plosila, J. (2013, September). LiRCUP: Linear regression based CPU usage prediction algorithm for live migration of virtual machines in data centers. In 2013 39th Euromicro Conference on Software Engineering and Advanced Applications (pp. 357-364). IEEE.
[5] Xue, J., Yan, F., Birke, R., Chen, L. Y., Scherer, T., Smirni, E. (2015, November). PRACTISE: Robust prediction of data center time series. In 2015 11th International Conference on Network and Service Management (CNSM) (pp. 126-134). IEEE.
[6] Ismaeel, S., Miri, A. (2019, January). Real-time energy-conserving VM-provisioning framework for cloud-data centers. In 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC) (pp. 0765-0771). IEEE.
[7] TFLearn: http://tflearn.org
[8] TensorFlow: https://www.tensorflow.org
[9] Siqi Shen, Vincent van Beek, Alexandru Iosup: Statistical Characterization of Business-Critical Workloads Hosted in Cloud Datacenters, CCGrid (2015)
[10] http://gwa.ewi.tudelft.nl/datasets/gwa-t-12-bitbrains