クラスタリングを用いた時系列データ回帰モデル化手法の提案

高橋 佑里子 (指導教員：小口 正人)

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全ての 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].

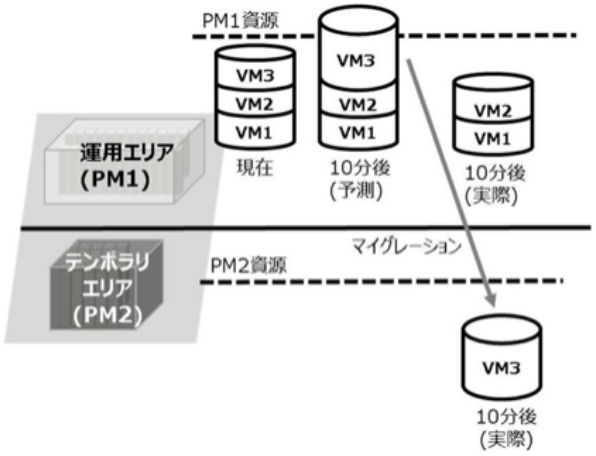

図 1: VM 制御のイメージ

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を提案する．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，なるべく少ない再学習での既存モデル回帰精度向上の施策を検討した．その結果，時系列データを学習に必要な長さごとに抽出し，それらをクラスタリングした結果を元にデータを選定することで，通常よりも少ない再学習でモデルの回帰精度が向上することが確認できた．

2 関連技術

本研究では，深層学習ライブラリとして TFLearn[3] を使用し，モデルには RNN を長期依存が可能かつ計算量が比較的少なくなるよう改良した GRU を採用した．モデルの再学習には，学習済みのモデルの上層部を一部再学習させるファインチューニングという手法を用いた．また，回帰精度の評価指標には RMSE を用いた．この値が小さく 0 に近いほど精度が良いということになる．

3 実験準備

3.1 実験に使用するデータ

本実験では，VM の CPU 使用率時系列データセットとして，Bitbrains IT Services Inc.[4][5] が公開しているものを使用した．前処理として，2000 個のデータセットに平滑化，正規化処理を行った後，実験で使用するデータセットを選択するための階層型クラスタリングを行った．クラスタ数を 20 としたときのデンドログラムは図 2 のようになった．縦軸はクラスタ間のユーグリッド距離，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．

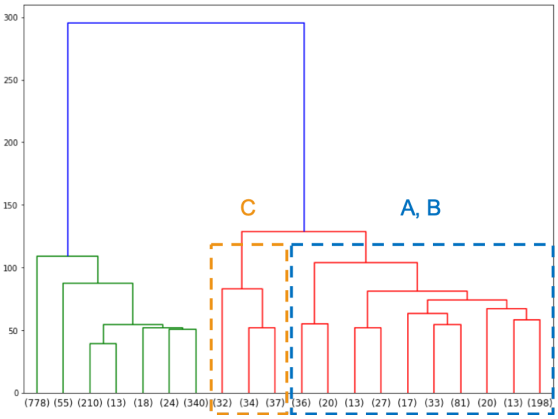

図 2: 全体の階層型クラスタリング結果と A,B,C の範囲

3.2 データ粒度の最適化

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，history 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 history 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，学習させる．そのため，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ことになる．そこで，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，それらをクラスタリングした結果を活用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．

ファインチューニング実験を行う前に，データを抽出する最適な長さを見つけるための history 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 history 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．この実験で，history が 24 のときに RMSE の平均値が最も小さくなったため，本実験では直前の 24 個のデータをもとに 1 個先のデータを

予測する深層学習ネットワークを構築することにした．そして，A,B,C に該当する 309 個のデータを，開始点を 1 点ずつずらしながら，学習元データ 24 点と正解データ 1 点を繋げた合計 25 点ごとに抽出した．

3.3 クラスタリングによるデータセット分析

3.2 節で抽出した細かい数多くの時系列データを，各環境ごとに k-means 法クラスタリングにより 10 個のクラスタに分類した．結果は図 3 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．

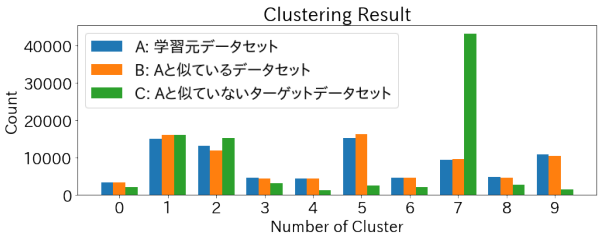

図 3: 細かいデータ抽出後の A,B,C の k-means 法クラスタリング結果

4 実験

上記のクラスタリング結果に基づいて，A で事前学習したモデルに C を使用してファインチューニングを 2 種類の方法で行った．方法 1 は，多く分類されたクラスタのデータを使用してファインチューニングを行うというもので，クラスタ No.1(C 全体の 18.0%)，No.2(C 全体の 16.9%)，No.7(C 全体の 48.2%) のデータを使用して行った．方法 2 は，C の各クラスタのデータを一定の割合で使用してファインチューニングを行うというもので，割合を 20%刻みで段階的に変更して行った．

そして，A で事前学習したファインチューニングを行う前のモデル，B で学習したモデル，方法 1,2 でファインチューニング行った後のモデル，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，それらの回帰精度を比較することで，適切なファインチューニングの方法を検討した．

5 実験結果と考察

実験結果は図 4 のようになった．縦軸は RMSE の平均値，横軸はファインチューニングのパターンである．グラフ左端の青色で示したものが A で学習した場合の結果，左から 2 番目のオレンジ色で示したものが B で学習した場合の結果，黄緑色で示したものが方法 1 の結果，赤色で示したものが方法 2 の結果，右端の緑色で示したものが C で学習した場合の結果である．

B の結果から，細かいデータ抽出後のクラスタリングで同じような分布になるデータセットで学習すると，精度はほとんど変わらないということが読み取れる．方法 1 の No.1, No.2, No.7 の結果からは，ターゲットデータセットであっても偏ったデータを使用してファインチューニングを行うことで，精度が低下するということが読み取れる．また，方法 2 の結果からは，ターゲットデータセットを 40%以上一定の割合で使用して

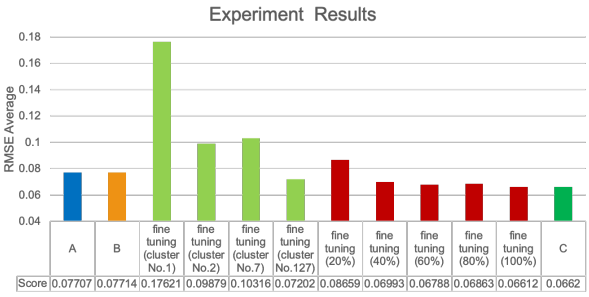

図 4: ファインチューニング実験結果

ファインチューニングを行うことで，ターゲットデータセット全てを使用して学習したモデルと同等な精度にまで向上するということが読み取れる．

再学習で使用するデータ数と精度の向上という点で考えると，方法 2 の 40%の場合が最も良いということが分かる．方法 1 の No.1,2,7(C 全体の 83.1%) の場合も精度は向上しているが，方法 2 の結果と比較すると，データ数と精度の両面で劣る．これより，クラスタ間のデータの分布傾向の違いを使ってファインチューニングを行うことが重要であることが分かる．

6 まとめと今後の予定

既存予測モデルの汎用化に向けて，なるべく少ないターゲットデータセットの再学習で回帰精度を向上させる方法を検討した．実験の結果，学習に必要な長さごとに抽出したターゲットデータセットをクラスタリングで分類し，各クラスタのデータを一定の割合で均等に使用してファインチューニングを行うという方法により，少量の再学習で十分に精度が向上することを確認した．

今後は，ファインチューニングを行う範囲を変えたり，新たなデータの選定方法を考えたりしながら，モデルの汎用化に向けてさらに実験を進めていきたい．回帰の次の段階として，予測の精度向上に向けた取り組みも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る．

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] TFLearn: http://tflearn.org

[4] Siqi Shen, Vincent van Beek, Alexandru Iosup: Statistical Characterization of Business-Critical Workloads Hosted in Cloud Datacenters, CCGrid (2015)

[5] http://gwa.ewi.tudelft.nl/datasets/gwa-t-12-bitbrains