

# 深層学習を用いたネットワークの輻輳予測に用いる特徴量に関する検討

明石季利子<sup>†</sup> 中尾 彰宏<sup>††</sup> 山口 実靖<sup>†††</sup> 小口 正人<sup>†</sup>

<sup>†</sup> お茶の水女子大学 〒112-8610 東京都文京区大塚 2-1-1

<sup>††</sup> 東京大学 〒113-8654 東京都文京区本郷 7-3-1

<sup>†††</sup> 工学院大学 〒163-8677 東京都新宿区西新宿 1-24-2

E-mail: <sup>†</sup>kiriko@ogl.is.ocha.ac.jp, <sup>††</sup>nakao@nakao-lab.org, <sup>†††</sup>sane@cc.kogakuin.ac.jp,  
<sup>††††</sup>oguchi@is.ocha.ac.jp

**あらまし** 突然発生する通信障害は、大規模災害時による通信過多や DDoS 攻撃、同時に起こる OS アップデートなど、様々な原因で引き起こされる。TCP によるデータ転送において、既に輻輳制御アルゴリズムが実装されている。しかし、輻輳によるスループットの低下はネットワーク機器依存の要因の他に、ユーザの行動が原因となるようなものも含まれ、それに対処するには従来の方法だけでは難しい。本研究では、ネットワークのトラフィックデータを用いて深層学習 LSTM モデルによる輻輳の時系列予測を行うことで、なるべく早く輻輳の検出をすることを目指す。加えて、有線通信時のパケット情報から、輻輳が起きることを高確率で予測できる特徴量を検討する。

**キーワード** 深層学習、LSTM、ネットワークトラフィック、輻輳予測

## A Study on Features Used for Congestion Prediction in Networks Using Deep Learning

Kiriko AKASHI<sup>†</sup>, Akihiro NAKAO<sup>††</sup>, Saneyasu YAMAGUCHI<sup>†††</sup>, and Masato OGUCHI<sup>†</sup>

<sup>†</sup> Ochanomizu University 2-1-1 Otsuka, Bunkyo-ku, Tokyo 112-8610 Japan

<sup>††</sup> University of Tokyo 7-3-1-1 Hongo, Bunkyo-ku, Tokyo 113-8654, Japan

<sup>†††</sup> Kogakuin University 1-24-2 Nishi-shinjuku, Shinjuku-ku, Tokyo 163-8677, Japan

E-mail: <sup>†</sup>kiriko@ogl.is.ocha.ac.jp, <sup>††</sup>nakao@nakao-lab.org, <sup>†††</sup>sane@cc.kogakuin.ac.jp,  
<sup>††††</sup>oguchi@is.ocha.ac.jp

## 1. はじめに

現代において、コンピュータシステムはビジネスや日常生活で欠かすことのできない存在となっており、通信サービスを提供する装置とその間を接続する通信ネットワーク及び関連設備に何らかの障害が発生すると広範囲に影響が生じる。突然発生する通信障害は、大規模災害時による通信過多や DDoS 攻撃、同時に起こる OS アップデートなど、様々な原因で引き起こされる。通信量が急激に増加し、ネットワーク混雑となる輻輳状態を回避するため、様々な通信障害の対策が取られてきた。TCP によるデータ転送において、既に輻輳制御アルゴリズムが実装されている。しかし、TCP に輻輳回避のアルゴリズムが備えられているにもかかわらず、輻輳によりスループットが著しく低下する現象が発生する。この現象は、ルータのメモリ不足やエンドノードでのネットワークの帯域よりも狭い帯域のリンクが転送経路上に存在するとき、処理の遅いルータがある

時などに見られる。また、トラフィックの輻輳にはユーザの行動が原因となるようなものも含まれ、それに対処するには従来の方法だけでは難しい。そこで本研究では、ネットワークのトラフィックデータを用いて深層学習による輻輳予測を行うことで、なるべく早く輻輳の検出をすることを目指す。加えて、有線通信時のパケット情報から、輻輳が起きることを高確率で予測できる特徴量を検討する。

## 2. 深層学習

### 2.1 RNN (Recurrent Neural Network)

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、系列が長くなると誤差逆伝播のアルゴリズムでは勾配の消失、発散などの問題が生じ、実際には 2, 3 ステップ前までの記憶しか保持できないという欠点がある。

## 2.2 LSTM (Long Short-Term Memory)

LSTM は、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改良したものである。入力ゲートと出力ゲートは、必要な誤差信号だけが適切に伝播するようにゲートの開閉を行うために導入され、忘却ゲートは、入力の系列パターンが変わったときに、一度学習した内容を忘れてノードの状態を一気に初期化するために導入された。LSTM の構造を図 1 に示す。本研究で扱うパケットデータは時系列データであり、比較的長いパターンを学習する必要があるため、長期の時系列データの学習を行うのに有効である LSTM を用いてネットワークトラフィックの予測を行なっている。

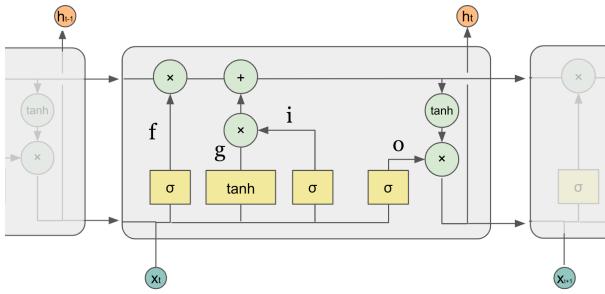


図 1: LSTM の構造

## 3. 関連研究

ネットワークの輻輳が発生すると、TCP/IP は TCP 輻輳制御を行う。TCP 輻輳制御はスロースタートと輻輳回避に基づいたアルゴリズムであり、TCP Tahoe, TCP Reno, TCP NewReno はこのアルゴリズムを用いて輻輳制御を行なっている [1] [2]。近年リリースされた Linux のバージョンでは、BIC TCP [3] や CUBIC TCP [4] のような改良された TCP 輻輳制御アルゴリズムが実装されている。Windows では、ロスペースと遅延ベースを組み合わせた輻輳制御方式である Compound TCP [5] が実装されている。さらに、Google 社が 2016 年にリリースした帯域遅延積に基づいた輻輳制御アルゴリズムである TCP BBR が注目されている [6]。[7] では、Rate-Adaptive TCP (RATCP) というアルゴリズムが開発されている。この TCP 輻輳制御システムは、ボトルネックレートフィードバックに応じて輻輳ウィンドウを変化させるシステムである。ここでは、TCP に対してよりよいフィードバックをするために、様々なネットワーク状況下で RATCP と TCP を比較している。このように、TCP 輻輳制御は長年にわたり異なる観点から改良されてきた。しかしながら、これらのアルゴリズムは、ネットワークの輻輳によるパケットロスのようなイベントが発生してから制御するシステムである。それに対し、輻輳の発生を事前に予測することができれば、非常に効率的なトラフィック制御が実現できることは明らかである。

## 4. 関連技術

### 4.1 Tensorflow

Tensorflow [8] は、2015 年に Google が開発したオープンソースのライブラリであり、機械学習やニューラルネットワークのさまざまなモデルやアルゴリズムを使うことができる。ニューラルネットワークの構築、訓練ができるシステムの要求に応え、処理に対してテンソルを扱っている。手書きの数字の認識、画像認識、自然言語処理などに利用される。本研究では Tensorflow を用いて、時系列データ学習モデル LSTM の構築・実験を行う。

### 4.2 iPerf

iPerf [9] は、NLANR/DAST によって開発された、ネットワークのスループットを測定するためのフリーソフトウェアである。サーバモードとクライアントモードの 2 種類の機能があり、サーバ、クライアント間でテストデータを流し指定した通信を発生させ、片方向または両方向の両端間のスループットの測定を行う。IP アドレス、ポート番号、プロトコル、送信するデータなど様々な条件を設定するためのオプションが用意されており、ネットワーク転送性能を測定することができる。そのため、スループット測定の他に障害発生時の原因調査などにも使用される。また、帯域を指定してトラフィックを発生させることができるため、ネットワーク負荷試験にも用いられるツールである。Windows や Linux など各種 OS で使用することができる。

### 4.3 カーネルモニタ

TCPにおいては、カーネル内部で定義された輻輳ウィンドウ (cwnd) などの輻輳制御パラメータを用いて輻輳制御を行なっている。カーネル内部の処理は通常バックグラウンドで進められているため、通常ユーザ空間からその処理の様子を監視することはできない。そこで既存研究 [10] によりカーネル内部の情報を見るためにカーネルモニタと呼ばれるシステムツールが開発された。カーネルモニタは、Android 端末の TCP 通信路における輻輳ウィンドウサイズや輻輳制御に用いられる閾値 (ssthresh) などのカーネル内部の様々なパラメータをリアルタイムにモニタする。TCP のソースコードにモニタ関数を挿入し、カーネルを再構築することで、メモリからログを得ることが可能となる。既存研究 [10] は、すでに Android が搭載されたスマートフォン端末やタブレット端末へのカーネルモニタの組み込みに成功しており、本研究においては同様の方法で Linux マシンへの導入を行なった。

## 5. 本研究で使用する機械学習手法と実験環境

本実験は、学習用のデータを iPerf 通信で取得して、機械学習モデルを使って予測を行い、評価する。輻輳状態の度合いを予測する回帰モデルと、輻輳発生の有無を予測する二値分類モデルを作成した。

### 5.1 機械学習手法と評価方法

本研究では、時系列データを扱うため LSTM モデルを用いて学習モデルを作成した。正解データは、回帰モデルでは輻輳状態のマーク数、二値分類モデルでは輻輳発生の有無とする。

予測精度の評価には、平均平方二乗誤差（RMSE）と混合行列を用いた。

## 5.2 iPerf を用いた通信実験

実験用マシン 1 台をサーバ、5 台をクライアントとして iPerf を用いた TCP 通信を行い、送受信されたパケットを各マシンでキャプチャする（図 2）。その際カーネルモニタを用いて通信時の幅轍ウインドウの値（cwnd）を取得する。TCP 幅轍制御アルゴリズムは Reno を用いた。

学習用のデータは、iPerf の通信のみを行った実験 1 と、別の通信を行なながら iPerf 通信を行なった実験 2 を用意した。

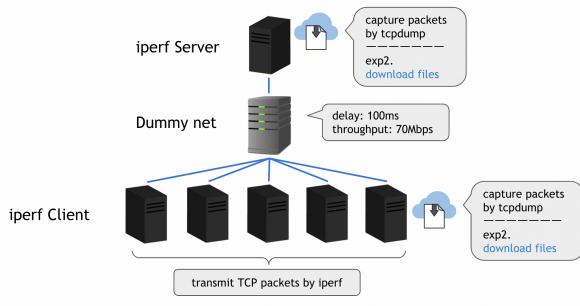


図 2: 実験環境

## 5.3 学習データ

入力データは  $t-9$  秒～ $t$  秒の 10 秒間に送受信したパケットから 1 秒おきに 3 つの特徴量を導出したものとしており、詳細は以下である。

- 1 秒間あたりのパケット総数
- 広告ウインドウの値（rwnd）
- 幅轍ウインドウの値（cwnd）

正解データは、時刻  $t+1$  秒の幅轍状態をマークしたパケットの数とする。本実験の幅轍状態とは、Wireshark の TCP analysis 機能によってマークした以下の 4 種類のパケットとする。

- 重複 ACK
- 高速再転送
- 重複受信
- ACK ロスによる再送パケット

学習データはクライアントから 1000 秒間連続でパケットを送信したものであり、学習のためにシーケンスデータに変換した。図 3、図 4 は、iPerf クライアントとサーバそれぞれで取得した学習データを 0～1 に正規化したグラフである。

学習に用いた計算機の性能を表 1、トライフィック発生に用いた計算機の性能を表 2 に示す。

表 1: 実験で用いた計算機の性能

OS	Ubuntu 16.04.2 LTS
CPU	Intel Xeon Gold 5115 CPU @2.40GHz
GPU	GeForce
Memory	32Gbyte

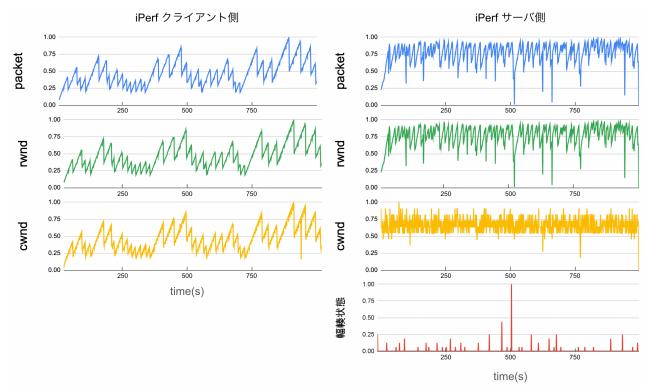


図 3: 実験 1 の学習データ（packet, rwnd, cwnd, 幅轍状態）

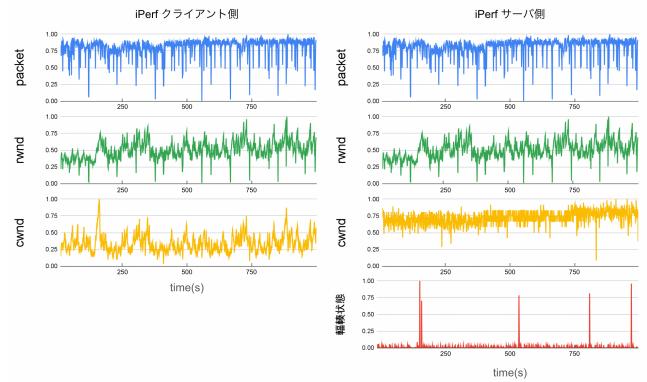


図 4: 実験 2 の学習データ（packet, rwnd, cwnd, 幅轍状態）

表 2: トライフィック発生に用いた計算機の性能

OS	Ubuntu 16.04.1 LTS
CPU	Intel Xeon CPU E3-1270 V2 @3.50GHz
Memory	16Gbyte

## 6. バリデーションデータを用いた幅轍予測

100 秒間のバリデーションデータを用いて幅轍予測を行なった。実験 1 では、iPerf 通信以外の通信を発生させず、iPerf サーバと iPerf クライアント間で発生させた通信のパケットキャプチャを行い、学習データとした。実験 2 では、別の通信を行ながら、iPerf 通信を行なったものを学習データとする。iperf コマンドの実行中に、iPerf サーバとクライアントでファイルのダウンロードを行なった。

### 6.1 学習モデルの作成

パケットデータの取得元をサーバ側（ServerSide）・クライアント側（ClientSide）・両端（BothEnds）の 3 パターンに分ける。また、特徴量の比較のため、パケット数・cwnd・rwnd・すべてを含む場合（all）の 4 パターンに分け、全 12 種類の学習モデルを作成した。

### 6.2 回帰モデルによる幅轍予測

評価指標は RMSE を使用する。RMSE は時刻  $t$  における予測値と正解値の誤差を表す。

実験 1 の予測結果（図 5）を見ると、おおまかな変動は予測可能であることがわかった。実験 2 の予測結果（図 6）は、特

微量の packet のモデルに比べて、全てを含む場合 all のモデルは変動を敏感に察知しているように見られる。しかし、RMSE の値（図 7）では、どのモデルもあまり差が見られないという結果となった。回帰モデルの予測において、RMSE は変動の高さの誤差が強調されて捉えられてしまう特徴がある。それによって、輻輳の振る舞いを予測できいていても評価がしづらく、モデル精度に差が見えづらい原因になり得る。したがって、輻輳予測を分類問題として予測する 6.3 章の二値分類モデルによる予測を行う。

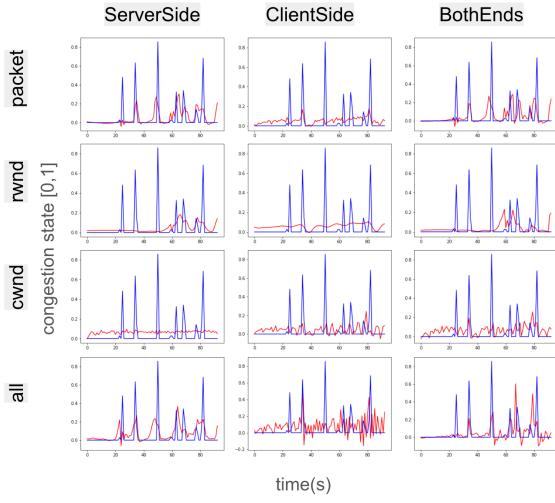


図 5: 実験 1・回帰モデルによる輻輳予測

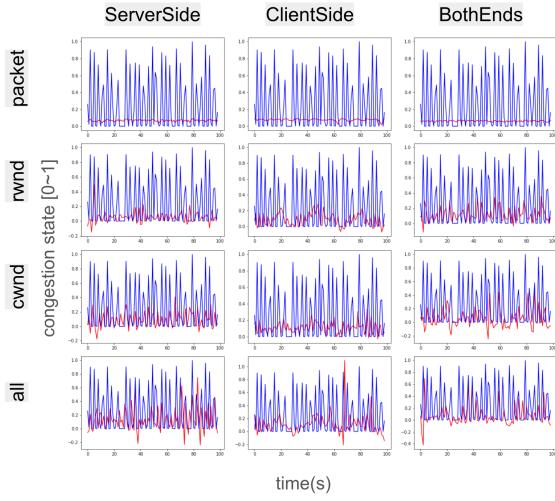


図 6: 実験 2・回帰モデルによる輻輳予測

### 6.3 二値分類モデルによる輻輳予測

連続的な数値予測ではなく、輻輳かそうでないかの二値分類モデルでバリデーションデータの予測を行った。

実験 1 の結果（図 8）では、ServerSide・cwnd のモデルは、正解値が輻輳状態を示す 1 のタイミングであるのに関わらず、連続して輻輳でないと予測している。ClientSide・packet,rwnd のモデルは、連続して輻輳を予測している。一方、BothEnds・cwnd のモデルは連続した予測ではないが、過度な輻輳判定が

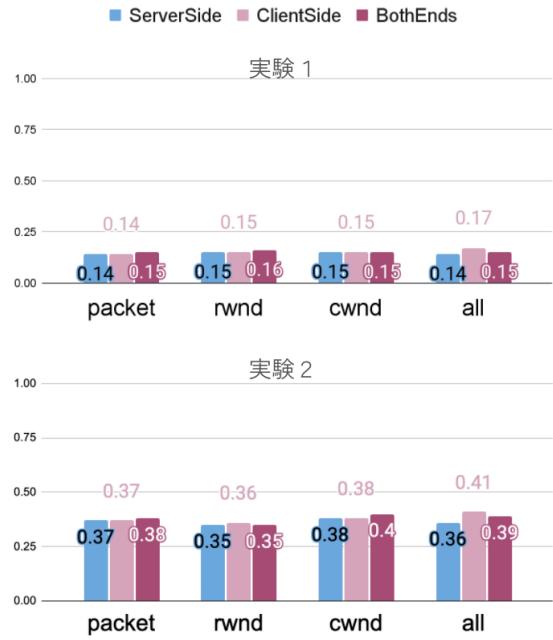


図 7: 回帰モデルの予測における RMSE の値

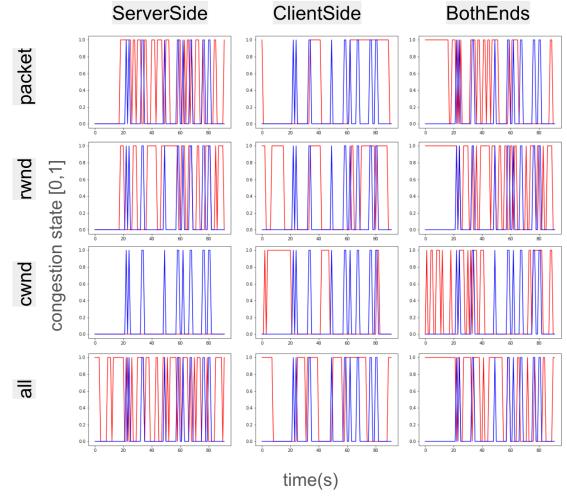


図 8: 実験 1・バリデーションデータの輻輳予測

多いことがわかる。

実験 2 のバリデーションデータを用いた予測結果を図 9 に示す。また、100 秒間のデータの詳細を見やすくした、始め 50 秒間データの結果を図 10 に示す。

## 7. 性能評価

### 7.1 混合行列

混合行列は、実際のクラスを縦軸、モデルが予測したクラスを横軸として表した行列である。輻輳状態である「Positive」もしくはそうでない「Negative」の 2 クラスに分類する場合の混合行列の各要素を表 3 に示す。

正解値と予測値の混合行列を基に、正解率・適合率（陽性と予測されたサンプルのうち正解したサンプルの割合）・再現率（実際に陽性のサンプルのうち正解したサンプルの割合）・F 値

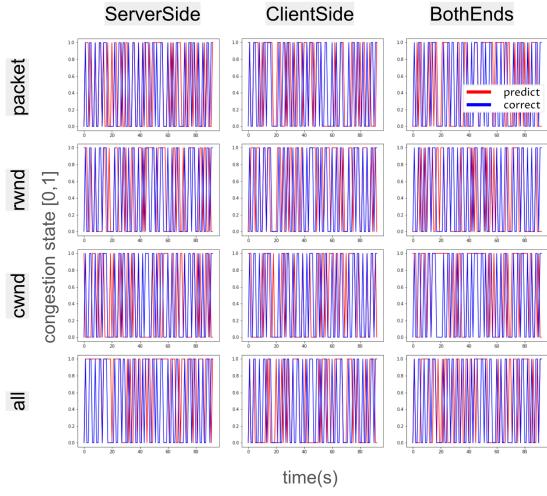


図 9: 実験 2・バリデーションデータの幅轍予測（100 秒間）

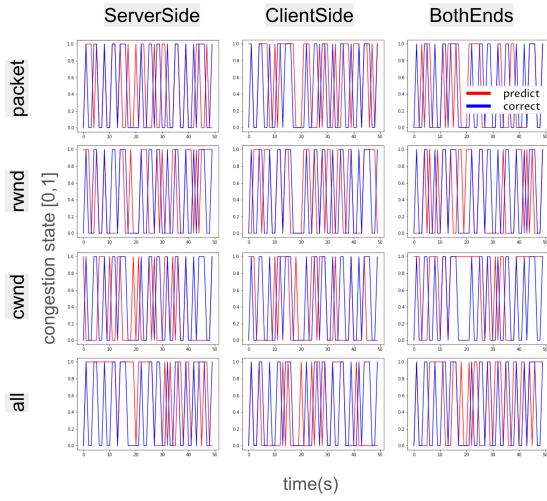


図 10: 実験 2・バリデーションデータの幅轍予測（50 秒間）

表 3: 混合行列

予測 実際	Positive	Negative
Positive	TP : 真陽性 (True Positive)	FN : 偽陰性 (False Negative)
Negative	FP : 偽陽性 (False Positive)	TN : 真陰性 (True Negative)

（適合率と再現率の調和平均）を算出する式をそれぞれ以下に示す。適合率の値が大きいほど、誤検知が少なく、再現率の値が大きいほど、検知漏れが少ないと判断できる。

- 正解率 =  $TP + TN / ( TP + TN + FP + FN )$
- 適合率 =  $TP / ( TP + FP )$
- 再現率 =  $TP / ( TP + FN )$
- $F$  値 =  $( 2 \times \text{適合率} \times \text{再現率} ) / ( \text{適合率} + \text{再現率} )$

## 7.2 結果と考察

### 7.2.1 実験 1 の性能評価

正解率が最も高いのは ServerSide・cwnd のモデルとなった（図 11）。しかし、適合率や再現率ともに 0 となっている。バリ

デーションデータを用いて予測した結果（図 8）を見ると、幅轍状態を予測しないことで幅轍でない状態を当てた数が多くなるため、正解率を高くしている。これは予測モデルとして意味を成さない。再現率、F 値どちらも、ClientSide・cwnd のモデルが高い値となった。次に F 値が高いのは Both・cwnd のモデルという結果となった。

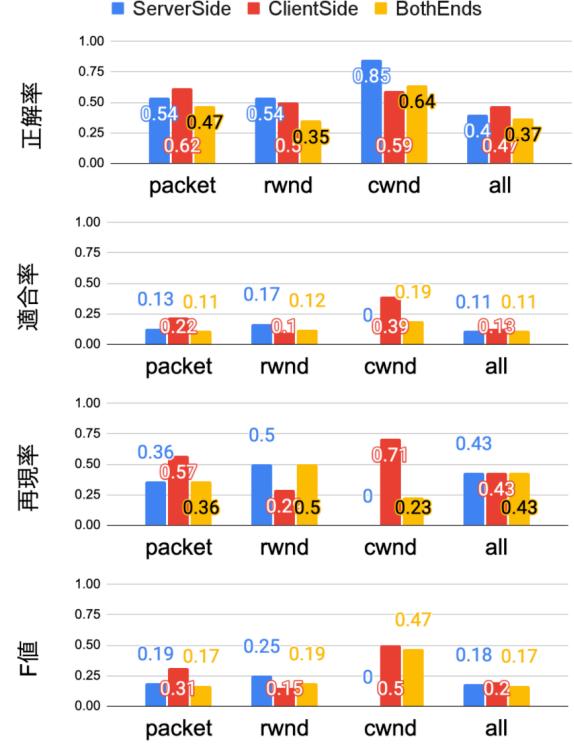


図 11: 実験 1・学習モデルの性能評価

### 7.2.2 実験 2 の性能評価

正解率が一番良いのは、ClientSide・cwnd のモデルとなった（図 12）。全体的に実験 1 の結果と比べて、他モデルと差がなくなりグラフに凹凸がなくなった。また、適合率と F 値が iPerf 通信のみの結果に比べて精度が上がっている。iPerf 通信のみの実験データに比べて、ファイルダウンロード時のデータを学習に用いた方が、良い予測ができることがわかる。再現率が高いのは、ServerSide・all のモデルとなった。サーバ側の幅轍を当てるため理にかなう結果であるといえる。次に ClientSide・rwnd のモデルの再現率が高く、F 値は最も高いモデルであった。

## 8. まとめと今後の課題

### 8.1 結論

本研究は、幅轍の早期検知を目指し、ネットワークトラフィックデータを用いて深層学習モデル LSTM による幅轍の時系列予測を行った。実験用サーバのモニタリングデータから取得した特徴量を用いて全 12 種類の予測モデルを作成した。幅轍が検知されない「偽陰性」を抑えることを評価軸とし、高確率で予測できる特徴量を検討した。再現率の観点から、最も偽陰性が低く幅轍予測に適しているモデルを検討した。結果は、実験 1 では ClientSide・cwnd のモデル、実験 2 では ServerSide・

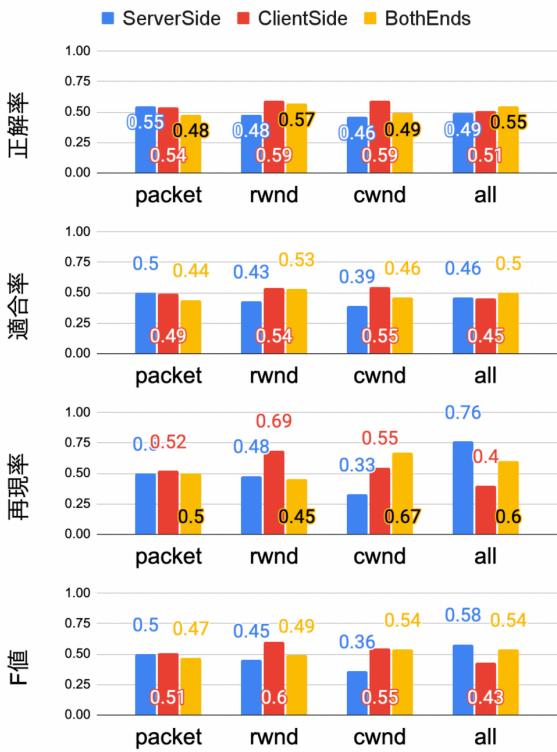


図 12: 実験 2・学習モデルの性能評価

all のモデルで高い予測精度が得られた。

## 8.2 今後の課題

幅轍でないことを予測し続けることで正解率を高めてしまう結果を踏まえて、再現率が高いモデルの正解率をある一定の精度高めることでより最適化されたモデルを作成できると考えている。そのため、陽性ラベルの精度が 50%以上であることを条件に、予測精度を最適化することを目標としてモデルの実験を行いたい。加えて、学習するトラフィックデータの他の特徴量を追加して比較実験を行うことで、幅轍検知と関連性が高い特徴量を発見したい。

## 文 献

- [1] M. Allman, V. Paxson, and E. Blanton. Tcp congestion control. In *Internet RFC 5681*, Sept. 2009.
- [2] T. Henderson, S. Floyd, and A. Gurtov. The newreno modification to tcp's fast recovery algorithm. In *Technical Report. IETF*, 2004.
- [3] L. Xu, K. Harfoush, and I Rhee. Binary increase congestion control for fast long-distance networks. In *Proc. INFOCOM*, Mar. 2004.
- [4] S. Ha, I. Rhee, and L. Xu. Cubic: A new tcp-friendly high-speed tcp variant. In [http://netsrv.csc.ncsu.edu/export/cubic\\_a\\_new\\_tcp\\_2008.pdf](http://netsrv.csc.ncsu.edu/export/cubic_a_new_tcp_2008.pdf), year.
- [5] K. Tan, J. Song, Q. Zhang, and M. Sridharan. A compound tcp approach for high-speed and long-distance networks. In *Proc. INFOCOM*, Apr. 2006.
- [6] Neal Cardwell, Yuchung Cheng, C. Stephen Gunn, Soheil Hassas Yeganeh, and Van Jacobson. title. In *title*, year.
- [7] Aditya Karnik and Anurag Kumar. Performance of tcp congestion control with explicit rate feedback. In *Proceedings of IEEE/ACM Transactions on Networking (TON) archive*

*Volume 13 Issue 1, 2005.*

- [8] Tensorflow. <https://www.tensorflow.org/>. (Accessed on 01/28/2021).
- [9] Nlanr – national laboratory for applied network research. <http://dast.nlanr.net/>. (Accessed on 01/04/2021).
- [10] Kaori Miki, Saneyasu Yamaguchi, and Masato Oguchi. Kernel monitor of transport layer developed for android working on mobile phone terminals. In *Tenth International Conference on Networks (ICN)*, 2011.