一般社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
IN2022-53(2023-01)

フローシーケンスによる侵入検知システムのドメイン適応能力の向上

グエンザロック†　渡部　康平†

† 長岡技術科学大学 大学院工学研究科　〒940–2188 新潟県長岡市上富岡町 1603–1
E-mail: †s203145@nagaokaut.ac.jp, †k_watabe@vos.nagaokaut.ac.jp

あらまし　ネットワーク侵入検知システム (NIDS) は，ネットワークに対する潜在的な脅威を特定するツールである．近年，侵入を効率的に検知するためのソリューションとして，機械学習（ML）アルゴリズムを用いた様々なフローベースの NIDS 設計が提案されている．しかし，従来の ML ベースの分類器は，ドメイン適応能力が低いため，実世界で広く採用されるには至っていない．本研究では，ネットワーク NIDS の領域適応能力を向上させるために，フローのシーケンスを利用する可能性を探ることを目的としている．本提案は，自然言語処理技術と，データをコンテキストに応じてモデル化するのに有効な BERT フレームワークを用いている．初期の実証実験の結果，本アプローチは従来のアプローチと比較して，ドメイン適応能力が向上していることが示された．提案手法は，ロバストな NIDS を構築するための新たな研究手法を提供する．

キーワード　ネットワーク, IDS, BERT, フローベース, 機械学習

Flow Classification Using Flow Collections and Deep Learning

Loc Gia NGUYEN† and Kohei WATABE†

† Graduate School of Engineering, Nagaoka University of Technology
Kamitomiokamachi 1603–1, Nagaoka, Niigata 940–2188, Japan
E-mail: †s203145@nagaokaut.ac.jp, †k_watabe@vos.nagaokaut.ac.jp

Abstract　This study aims to explore the possibility of using collections of flows to improve classification accuracy of network flow data. The proposal uses machine learning techniques, which is effective for modeling of data. Initial empirical results show that the proposed approach has improved classification accuracy compared to previous approaches. The proposed method provides a new methodology for building classification systems.

Key words　Network, IDS, BERT, Flow-based, Machine learning

1. はじめに

日常生活の多くがコンピュータネットワークに依存するようになり，システム内の情報を保護し，サービスの可用性を維持することが必要になってきた．金融取引や個人情報を扱うシステム，ソーシャルプラットフォームなどは，金融資産や社会への影響力を得ようとする攻撃者の格好のターゲットとなる．現在，これらの脅威に対抗するため，さまざまなサイバーセキュリティ対策が行われているが，侵入検知はその重要な要素の一つである．

侵入検知は，コンピュータシステムやネットワークで発生するイベントを監視し，悪意ある活動の兆候を分析するプロセスである．ネットワークトラフィックは，パケットベースまたはフローベースのフォーマットでキャプチャされる．パケットベースのデータには，完全なペイロード情報が含まれている．フローベースのデータは，パケットベースのデータを集約したもので，通常，ネットワーク接続からのパケットとメタデータの集合の様々な特性を含んでいる [1]．侵したがって，ネットワーク侵入検知システム (Network Intrusion Detection System; NIDS) は，パケットベースとフローベースの二つに分けることができる．前者は個々のパケットのペイロードやヘッダー情報を分析し，後者はパケットの集まりの集約された特性のみを分析する．フローベースの NIDS は，ネットワークトラフィックを分析する速度が速いため，最新のネットワークセキュリティシステムに適したソリューションとなっている．

ユーザの行動の変化や新しい攻撃の出現は，フローの特徴量の分布の変化として現れる．そのため，NIDS は悪意のあるフローを検知しつつ，新たな良性フローに対応する必要がある．フローベースの NIDS は，K-Nearest Neighbors や Random Forest などの機械学習アルゴリズムを用いてパケット収集の特性を分析する．しかし，これらの機械学習アルゴリズムは，実環境でよく見られるデータ分布の変化への適応性が低い場合が多く，

その結果，有用性が制限されている [2] [3]．NIDS のドメイン適応能力を向上させるために，Camila らは Energy-based Flow classifier (EFC) を提案した．これは，NIDS が良性フローの統計分布をモデル化し，この分布から外れたフローを悪性としてクラス化する統計的アプローチである．EFC は，NIDS のドメイン適応能力を向上させるが，特定のデータ分布にはまだ限界がある．分類器はフロー内の特徴量分布しかモデル化できないため，入力データとして単一のフローを用いることが，現在のアルゴリズムの限界の理由であると推測される．

本稿では, 時間的に近接したフローの集合であるシーケンスと，Bidirectional Encoder Representations from Transformers (BERT) フレームワークを用いて，ネットワークフローのシーケンスを表現することを提案する．そして，Multilayer Perceptron（MLP）がこのベクトル表現を処理し，フローの分類結果を生成する．我々の知る限り，これは NIDS のドメイン適応能力を向上させるために BERT を使用した最初の例である．

本研究の貢献度は以下の通りである．

- ネットワーク NIDS のドメイン適応能力を向上させるために，一連のネットワークフローからの情報を利用する可能性を検討する．
- 特徴抽出に BERT，分類に MLP を用いた NIDS を提案する．を用いた NIDS を提案する．
- 実と模擬トラヒックを用いて提案手法の領域適応能力を評価する．

本稿の構成は以下の通りである．第２章では，フローベース NIDS に関する文献のうち，著名な研究の概要を紹介する．第３章では，ネットワークフローとネットワーク NIDS の基本概念，本研究で使用するデータセット，および BERT フレームワークの紹介を行う．第４章では，本提案とその評価方法について詳細を説明する．第５章では，提案の性能評価結果と関連する議論を行う．最後に，第６章で本稿の結論を述べる．

2. 関 連 研 究

NIDS の実装に機械学習を用いることは広く研究されており [4]，フローデータからの特徴抽出に深層学習モデルを用いることで，機械学習手法をさらに改善する手法が採用されている．NIDS における特徴抽出の手法としては，オートエンコーダーが有名である．これは，最適な特徴を学習することで，出力をできるだけ入力に近づけるという考え方で動作する．Shone ら [5], Yan ら [6], Al-Qatf ら [7] は，オートエンコーダーがフローに関する重要な情報を保持しながら，その次元を縮小できることを示した．このアプローチにより，機械学習分類器に必要な処理時間を短縮し，分類能力を向上させることができる．Andresini ら [8] は，フローの種類ごとに異なるオートエンコーダーを学習し，これらのオートエンコーダーが生成する表現を使用して分類器を学習する．Khan ら [9] は，フローの特徴を入力とし，スコアを返すスコアリング関数としてもオートエンコーダーが利用できることを示した．この場合のオートエンコーダーは，正常なフローには低いスコアを，疑わしいフローには高いスコアを割り当てるように最適化されている．

表 1　CIDDS-001 と CIDDS-002 のフロー特徴

Date first seen	開始時間 フローが最初に見られた時間
Duration	フローの継続時間
Proto	トランスポートプロトコル（ICMP，TCP，UDP など）
Src IP	送信元 IP アドレス
Src Pt	送信元ポート
Dst IP	送信先 IP アドレス
Dst Pt	送信先ポート
Packets	送信パケット数
Bytes	送信バイト数
Flags	TCP フラグ

分類のためのモデルは，それを学習させるためのデータと同様に重要である．NIDS の学習用データの準備も NIDS の研究の重要な側面である．Verma ら [10] は，ラベル付きフローベースの CIDDS-001 データセットの統計解析を，k-nearest neighbor 分類と k-means クラスタリングアルゴリズムを使って行った．Abdulhammed ら [11] は, 不均衡な学習データを扱い, NIDS の性能を向上させるためのいくつかの手法を提案した．CIDDS-001 におけるサンプリング手法の有効性を，ディープニューラルネットワーク，ランダムフォレスト，投票，変分オートエンコーダ，スタッキング機械学習分類器を通して研究し，実験的に評価した．

先に紹介した論文はいずれも悪意のある活動の検出向上に焦点を当てたものであり，データ分布の変化に関するアプローチの性能は評価していない．Camila ら [12] は，機械学習のアプローチはデータ分布の変化に適応しにくいことを述べ，EFC を提案した．このアルゴリズムは，逆統計に基づき，良性フローに基づく統計モデルを推論する．著者らは，このアルゴリズムが異なるデータ分布に適応できることを示し，ゼロデイ攻撃の検知に適していることを示した．

3. 背　　景

本節では，まず，本研究の対象であるネットワークトラフィックフローデータの概念について紹介する．次に，NIDS の性能評価に使用するデータセットを紹介する．最後に，BERT のフレームワークの概要を説明する．

3.1 データセット

ネットワークフローとは，二つのホスト間で情報を伝達するパケットのシーケンスで，パケットには共通の性質がある．フロー内のすべてのパケットは，同じ 5 つのタプル（Src IP, Src Pt, Dst IP, Dst Pt, Proto）を共有する．パケット間の時間はすべて，任意のフロー満了タイムアウト値以下である．5 タプルまたは 3 タプル以外に，フローは送信パケット数，バイト数など他のフローキーを含むことができる．フローデータのフォーマットには，NetFlow や OpenFlow など多くの規格がある．

3.2 フローデータ

CIDDS-001 [13] と CIDDS-002 [14] は比較的最近の侵入検知ベンチマークデータセットで，一方向性の NetFlow データを含んでいる．CIDDS-001 データセットには二つのフローセットが含まれており，一つはシミュレーションされた OpenStack 環境

内でキャプチャされ，もう一つはインターネット上に配置されたサーバーからキャプチャされる．CIDDS-002 データセットには，シミュレートされた OpenStack 環境からのフローが 1 セット含まれている．

この研究では，CIDDS-001 OpenStack，CIDDS-001 External Server，CIDDS-002 OpenStack 環境を使用して，本提案のドメイン適応能力を評価した．CIDDS-001 のデータセットでは，模擬 OpenStack 環境から外部サーバ環境への変更は，環境間でフローの特徴分布が異なるため，ドメイン変更とみなされる．同様に，CIDDS-001 から CIDDS-002 の OpenStack 環境への変更もドメイン変更とみなされる．これらのドメイン変更を利用して，本提案のドメイン適応能力を評価する．

CIDDS-001 と CIDDS-002 内のネットワークフローは，すべてラベリングされている．CIDDS-001 と CIDDS-002 の OpenStack 環境では，良性のフローは normal，悪性のフローは攻撃の種類に応じて dos，scan，portScan，pingScan，bruteForce というラベルを付けている．外部サーバー環境には，OpenStack 環境からのシミュレーションフローと，インターネットからの実フローの両方が含まれている．実際のフローは，ポート 80 と 443 へのトラフィックに unknown，残りのトラフィックに suspicious というラベルを付けている．このとき，unknown と表示されたフローは一般ユーザからのトラ フィックである可能性が高く，本研究では良性とみなし，suspicious と 表示されたフローは悪性とみなした．

CIDDS-001 と CIDDS-002 のデータセットに含まれる属性の概要を表 1 に示す．送信元 IP アドレス，送信先 IP アドレス，Date first seen は任意であり，有用な情報を持たないため，本研究では分類に使用しなかった．

3.3 BERT

BERT は，Google が開発した変換器を用いた自然言語処理用の機械学習技術である．BERT フレームワークは，事前学習と微調整の二つのステップで構成されている． 事前学習では，BERT モデルはラベル付けされていないデータで学習する．微調整では，事前に学習したパラメータを使用してモデルを初期化し，その後，下流タスクからのラベル付きデータを使用して学習する．BERT は，二つの教師なしタスクで事前学習される．Masked Language Modeling (MLM) と Next Sentence Prediction (NSP) である． MLM では，文中のいくつかの単語が異なるトークンに置き換えられる．目的は，文中の他のマスクされていない単語に基づいて，マスクされた単語の元の値を予測することである．NSP では，BERT は文のペアを入力とする．目的は，ペアの 2 番目の文が文書内の次の文であるかどうかを予測することである．微調整のために，タスク固有の入力と出力が事前に訓練された BERT モデルに追加される．

4. 提 案 手 法

本節では，まず，フローシーケンスを NIDS の入力として使用することの利点を強調する．次に，提案するシステム構造の詳細について説明する．最後に，本研究で使用した評価方法について詳しく説明する．

表 2　フローシーケンス

Duration	Proto	Src Pt	Dst Pt	Packets	Bytes	Flags	Class
9.555	**TCP**	**54731**	**22**	**15**	**2163**	**.AP.SF**	**suspicious**
9.555	**TCP**	**22**	**54731**	**19**	**3185**	**.AP.SF**	**suspicious**
10.412	**TCP**	**22**	**57489**	**19**	**3185**	**.AP.SF**	**suspicious**
10.412	**TCP**	**57489**	**22**	**15**	**2163**	**.AP.SF**	**suspicious**
0	UDP	56475	19	1	46	.....	suspicious
0	ICMP	0	3.3	1	57	.....	suspicious
0.077	TCP	8000	52253	7	702	.AP.SF	normal
0.077	TCP	52253	8000	6	586	.AP.SF	normal
526.089	**TCP**	**22**	**59862**	**178**	**24253**	**.AP.SF**	**normal**
526.089	**TCP**	**59862**	**22**	**180**	**13471**	**.AP.SF**	**normal**
0	TCP	53213	23	1	46	....S.	suspicious
0	TCP	23	53213	1	40	.A.R..	suspicious

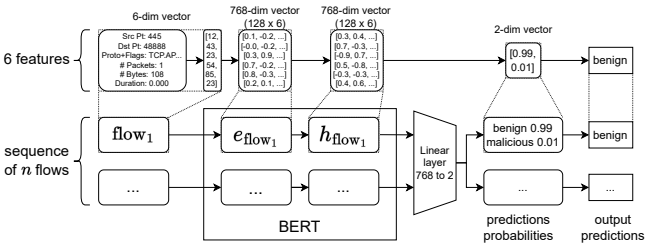

図 1　提案モデル

4.1 フローシーケンスからのコンテキスト

フローシーケンスとは，時間的に近くに出現するフローのことである．過去のフローをもとに，ネットワーク上で次に出現するフローの出現確率を決定することができる．表**??**は，フローシーケンスから得られる情報をもとに，フローが悪性か良性かを識別できる二つのフローシーケンスを示している．太字のフローはポート番号とフラグが類似しているが，最初のシーケンスでは悪意があり，2 番目のシーケンスでは良性である．これらのフローをその特徴だけで分類すると，これらのフローはすべて良性または悪性のいずれかに分類されると見込まれる．最初のシーケンスでは，これらのフローは連続して複数回出現し，ポート番号が急速に変化している．これは，攻撃者がポート 22 で攻撃を再試行しているパターンであると考えられる．2 番目のシーケンスでは，フローは孤立しているように見える．フローの特徴がほぼ同一である場合，対象フローを取り囲むフローを調べることで得られる余分な情報が，フローを区別するのに役立つのである．フローのシーケンスで生じるパターンは，特徴にとらわれない．つまり，同じパターンが任意の特徴で出現する可能性がある．したがって，フローシーケンスから得られる情報は，フロー内の特徴が異なる可能性のある異なるドメイン間でロバストである．つまり，フロー列を利用することで，フロー内の特徴に依存しない分類が可能になる．

4.2 機械学習モデル

我々は，BERT モデルと MLP モデルに基づくネットワーク侵入検知手法を提案する．BERT モデルは特徴抽出器として機能し，ネットワークトラフィックフローをベクトルに変換する．一方，MLP モデルは，ネットワークトラフィックフローを良性フローと悪性フローに分類する分類器として機能する．

まず，ネットワークトラフィックフローを自然言語と同様の

表 3　CIDDS 環境内のラベル

CIDDS-001 OpenStack		CIDDS-001 External Server		CIDDS-002 OpenStack	
ラベル	#	ラベル	#	ラベル	#
normal	28051906	*normal*	134240	*normal*	15598543
dos	2959027	*unknown*	77923	*scan*	562640
portScan	265918	*suspicious*	437911		
pingScan	6090	*portScan*	18719		
bruteForce	4992	*bruteForce*	2448		

表 4　実験で使用したデータのラベルの数

	CIDDS-001 internal	CIDDS-001 balanced	CIDDS-001 external	CIDDS-002
Benign	28051906	3236027	212163	15598543
Malicious	3236027	3236027	459078	562640
Total	31287933	6472054	671241	16161183

構造に整理し，フローを単語，フローのシーケンスを文として扱う．提案モデルの学習は二つのステップで構成される．まず，MLM タスクと良性フローのみを用いて，BERT モデルを事前学習する．フローには固有の順序がないため，一般的な BERT フレームワークで使用される位置符号化や特別な開始・終了トークンを使用しないことを選択する．次に，MLM の出力層を MLP の分類器に置き換え，微調整を行いる．このステップでは，データセット内のすべてのフローとそのラベルを使用して，BERT モデルと MLP 分類器を学習させた．シーケンス内のフローの分布を保持することは重要であるため，学習データセット内のフローは混ぜ合わせない．両ステップとも，学習基準としてクロスエントロピーを使用する．これは，第 1 ステップのコンテキストと第 2 ステップの正しいラベルが与えられたときに，正しいフローの特徴を推測する尤度を最大化するものである．

システムの全体的な構造を図 1 に示す．各フローは 6 つの特徴を含み，これらの特徴は数字 (flow_i) として符号化される．BERT は各数値を 128 次元のベクトルにデコードし，それらを連結して 768 次元のベクトル (e_{flow_i}) とし，フローを 768 次元の別のベクトルで表現する処理を行う (h_{flow_i})．BERT の出力は，MLP 分類器（ソフトマックス出力の線形層）に渡され，768 次元から 2 次元に縮小される．この 2 次元ベクトルは，各クラス（良性および悪性）の確率を表している．最終的な予測ラベルは，より高い確率を持つクラスが選択される．

5. 評　　価

5.1 実 験 条 件

本提案と，より優れたドメイン適応性を目指す EFC との比較を行う．この比較に含まれる他の ML ベースの NIDS 手法は, Decision Tree (DT), K-Nearest Neighbors (KNN), Multilayer Perceptron (MLP), Naive Bayes (NB), Support Vector Machine (LinSVM), AdaBoost (AB) and Random Forest (RF) である．ML ベースの分類器は，デフォルトの scikit-learn (version 1.1.2) の構成で展開した．提案モデルは PyTorch を用いて作成した．ほとんどの ML アルゴリズムは連続特徴より離散特徴の方が性能が良いため，全ての分類器に対して流れの特徴を離散化した．離散化の閾値は EFC の論文 [12] に記述されているものを使用した．表 5 に離散化の閾値の詳細を示す．

表 5　特徴離散化の閾値

特徴	閾値
Duration	0.001, 0.002, 0.003, 0.004, 0.005, 0.006, 0.01, 0.04, 1, 10, 100, ∞
Protocol	TCP, UDP, GRE, ICMP, IGMP
Src Pt	50, 60, 100, 400, 500, 40000, 60000, ∞
Dst Pt	50, 60, 100, 400, 500, 40000, 60000, ∞
Bytes	50, 60, 70, 90, 100, 110, 200, 300, 400, 500, 700, 1000, 5000, ∞
Packets	2, 3, 4, 5, 6, 7, 10, 20, ∞
Flags	$\{(f_0, f_1, f_2, f_3, f_4, f_5) \mid f_i \in \{0, 1\}\}$

三つの異なる環境 (ドメイン) のデータを使用する．CIDDS-001 OpenStack，CIDDS-001 External Server，CIDDS-002 OpenStack の三つの環境（ドメイン）のデータを用いて，他の分類器と比較して，本提案のドメイン適応能力を評価する．この分類器は，バイナリ分類タスクである異常検知でテストされる．そのため，ラベルは良性と悪性に分類される．各環境内のラベルは表 3 のようになる．normal と unknown のラベルが貼られたフローは良性，それ以外のラベルが貼られたフローは悪性と判断される．実験に使用したデータセットの構成を表 4 に示す．

本提案の領域適応能力を他の分類器に対して評価するため，二つの実験を行った．最初の実験では，CIDDS-001 OpenStack 環境のフローデータ全体に対して CIDDS-001 internal データセットで学習を行い，CIDDS-001 External Server 環境に対して CIDDS-001 external データセットで，CIDDS-002 OpenStack 環境に対しては CIDDS-002 データセットでテストを実施した．学習データセットのラベル不均衡は分類器の性能を低下させること [3] があるため，学習データセットをラベル均衡させた 2 回目の実験を行った．2 回目の実験では，1 回目の実験と同じデータセットでテストを行った．また，学習は CIDDS-001 OpenStack 環境のフローデータのサブセットを用いて行った．このサブセットは，CIDDS-001 balanced と呼ばれ，CIDDS-001 OpenStack 環境から悪意のあるフローをすべて残し，良性のフローをその数に合わせて減らすことによって作成されている．良性フローは，元のデータセットからのフローの順序を維持する方法でサンプリングされた．

5.2 指　　標

提案手法の性能は，NIDS の研究で広く利用されている Accuracy，F1-score，Recall，Precision を用いて測定する [4]．これらの指標は，TP (True Positive，すなわち悪意あるトラフィックが悪意あるものとして分類される), TN (True Negative，すなわち良性トラフィックが良性として分類される), FP (False Positive，すなわち良性トラフィックが悪意あるものとして分類される) および FN (False Negative，すなわち悪性トラフィックが良性として分類される) 値から計算される．このため，このようなトラフィックが発生する可能性がある．最初の指標である Accuracy は，フロー総数に対する正しく分類されたフローの比率として定義される．

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{1}$$

表 6　実験結果：CIDDS-001 internal で学習，CIDDS-001 external でテスト

	CIDDS-001 internal で学習 CIDDS-001 external でテスト			
分類器	Accuracy	F1-score	Recall	Precision
Proposal	**0.9078**	**0.9311**	0.9120	0.9511
EFC	0.8659	0.9044	**0.9278**	0.8822
DT	0.8491	0.8800	0.8088	0.9649
KNN	0.7978	0.8270	0.7067	**0.9967**
LinSVM	0.6784	0.6940	0.5333	0.9933
MLP	0.4380	0.3254	0.1982	0.9087
NB	0.3161	0.0000	0.0000	0.9937
AB	0.4606	0.3812	0.2430	0.8845
RF	0.8177	0.8510	0.7613	0.9647

Precision は，悪意のあるフローとして分類されたすべてのフローに対する，正しく分類された悪意のあるフロー フローと，悪意のあるフローに分類されたフローの比率として定義される．

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2}$$

Recall は，すべての悪意のあるフローに対して正しく分類された悪意のあるフローの比率として定義される．

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3}$$

二つ目の指標である F1-score は，Precision と Recall の調和平均値である．つまり，システムの再現性と精度の両方を考慮し，システムの精度を調べる統計的手法である．

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

5.3 実 験 結 果

BERT モデルとして，1 層，1 注目ヘッド，隠れサイズ 768 の BERT を実装した．学習時，バッチサイズ 512，シーケンス長 128 フロー（1 イテレーションで 65536 フローに相当）を使用する．Adam オプティマイザを用い，学習率 10^{-5} を一定とする．どちらの実験でも，MLM タスクで BERT モデルを 400 イテレーション訓練し，次に BERT モデルのパラメータを凍結して MLP 分類器のみを 1100 イテレーション微調整し，最後に BERT モデルの凍結を解除して BERT と MLP の両方を 400 イテレーション訓練している．合計で，2000 イテレーションでモデルを訓練した．テスト時には，シーケンス長を 1024 フローに増加させる．

表 6 と表 7 は最初の実験の結果であり，CIDDS-001 internal に対して学習が行われた．提案手法は CIDDS-001 external と CIDDS-002 test set の両方で精度 (0.9078 と 0.9913) と F1 スコア (0.9311 と 0.8578) に関して最高の性能を示した．EFC は CIDDS-001 外部テストセットにおいて，精度（0.8659），F1 スコア（0.9044）ともに 2 位であった．しかし，CIDDS-002 テストセットでは，EFC は精度 (0.9084) で最も悪く，F1 スコア (0.3317) で 3 番目に悪い結果であった．

表 7　実験結果：CIDDS-001 internal で学習，CIDDS-002 でテスト

	CIDDS-001 internal で学習 CIDDS-002 でテスト			
分類器	Accuracy	F1-score	Recall	Precision
Proposal	**0.9913**	**0.8578**	**0.7531**	**0.9962**
EFC	0.9084	0.3317	0.6534	0.2223
DT	0.9880	0.7948	0.6655	0.9864
KNN	0.9879	0.7924	0.6656	0.9789
LinSVM	0.9503	0.1042	0.0830	0.1400
MLP	0.9867	0.7722	0.6479	0.9554
NB	0.9638	0.0006	0.0003	0.0072
AB	0.9837	0.7148	0.5873	0.9131
RF	0.9881	0.7961	0.6670	0.9871

表 8　実験結果：CIDDS-001 balanced で学習，CIDDS-001 external でテスト

	CIDDS-001 balanced で学習 CIDDS-001 external でテスト			
分類器	Accuracy	F1-score	Recall	Precision
Proposal	**0.8581**	**0.9002**	**0.9358**	0.8673
EFC	0.8566	0.8985	0.9278	0.8710
DT	0.8496	0.8805	0.8098	0.9646
KNN	0.8128	0.8434	0.7374	**0.9850**
LinSVM	0.8123	0.8631	0.8650	0.8612
MLP	0.6671	0.6910	0.5441	0.9463
NB	0.2279	0.0282	0.0164	0.1013
AB	0.4742	0.4268	0.2862	0.8386
RF	0.8357	0.8679	0.7893	0.9639

表 9　実験結果：CIDDS-001 balanced で学習，CIDDS-002 でテスト

	CIDDS-001 balanced で学習 CIDDS-002 でテスト			
分類器	Accuracy	F1-score	Recall	Precision
Proposal	0.9870	0.7717	0.6315	**0.9921**
EFC	0.9088	0.3327	0.6534	0.2232
DT	0.9874	0.7871	0.6676	0.9586
KNN	0.9874	0.7867	**0.6698**	0.9532
LinSVM	0.5388	0.0260	0.1767	0.0140
MLP	0.9865	0.7832	0.7021	0.8854
NB	0.7908	0.0007	0.0021	0.0004
AB	0.9817	0.7170	0.6653	0.7774
RF	**0.9875**	**0.7891**	0.6692	0.9613

表 8 と表 9 は 2 番目の実験の結果であり，CIDDS-001 balanced に対して学習が行われた．提案手法は CIDDS-001 external に

対して，精度 (0.8581) と F1 スコア (0.9002) の点で最高の性能を示すことがわかった．CIDDS-002 では，RF が精度 (0.9875) と F1 スコア (0.7891) の点で最も良い性能を示した．EFC は，CIDDS-001 外部では強い性能を示したが，CIDDS-002 では劣勢であった．CIDDS-001 external は悪意あるフローにやや偏っているため，より良性のフローを含む不均衡なデータセットと比較して，均衡のとれたデータセットで学習させると，CIDDS-001 external における DT，KNN，LinSVM，MLP の精度と F1 スコアがわずかに改善されることがわかった．CIDDS-002 テストセットでは，良性フローに偏っているため，同じ効果は観察されず，誤検出率が低く，分類器に有利である．

CIDDS-001 internal に含まれる余分な良性フローは，CIDDS-001 balanced と比較して，EFC，DT，KNN，AB，RF の性能に影響を与えない．しかし，提案手法の精度や F1 スコアは改善された．提案手法は，CIDDS-001 internal で学習させた場合に，最も大きな効果が得られた．これは，余分な良性フローが BERT モデルにより多くのコンテキスト情報を提供し，良性フローを識別する能力を向上させる一方で，分類結果がより多くのサンプルを持つクラスに対して偏ることがないためであると推測される．

提案手法は，DT，RF，KNN とともに，二つの異なるテストセットにおいて性能を維持することができ，優れたドメイン適応能力を示している．DT, RF, KNN の結果は，先行研究で報告されている良好な性能を反映している [5] [10], 提案手法は，CIDDS-001 内部で学習したテストデータセットと，CIDDS-001 外部で学習したテストデータセットで，他の全ての分類器より優れた性能を発揮することができた．

実験結果から，EFC は CIDDS-001 external（EFC の原著論文で使用されたものと同じデータセット）では良い性能を示すが，CIDDS-002（EFC の原著論文では使用されていない）の F1 スコアは著しく悪いことが分かる．そこで，CIDDS-002 からサンプリングした小規模な均衡テストセット（良性フローと悪性フローそれぞれ 10000 個ずつ）を用いて実験を行った [15]．その結果，EFC は DT，RF，KNN に匹敵する結果を得ることができた．また，Camila らが報告したように，EFC は小規模で均衡のとれたテストセットでは良好な性能を示すが，良性フローに大きく偏った不均衡なテストセット（本研究で用いた CIDDS-002 テストセット）では性能が大きく劣化することが分かった．

6. おわりに

我々は，フローシーケンスを入力とし，BERT モデルを利用する NIDS を提案した．従来の ML ベースの NIDS は，単一のフローからの情報しか利用しないため，ドメイン適応能力が低いと理論付けている．提案モデルの学習は，二つのステップで構成されている．ステップ２では，ラベルなしデータを用いて，コンテキストからフローを予測する BERT モデルを学習させる．ステップ２では，MLP 分類器を BERT モデルの出力に追加し，ラベル付きデータで学習させる．CIDDS-001 と CIDDS-002 における初期の実験結果では，提案手法は異なるドメイン間で一貫した結果を達成することが示された．

本研究では，良性フローと悪性フローの分類に MLP 分類器を使用したが，この分類器は，モデルの学習にラベル付きフローを必要とする．今後は，良性フロー列の分布をモデル化し，この分布からの逸脱は異常フローの可能性が高いという統計的アプローチによるフロー分類の実験を行う予定である．

7. 謝　辞

本研究の一部は科研費 20H04172 の助成を受けたものである．

文　献

[1] M. Ring, S. Wunderlich, D. Scheuring, D. Landes, and A. Hotho, “A survey of network-based intrusion detection data sets,” Computer Security, vol.86, pp.147–167, 2019.

[2] H. Li, Z. Chen, R. Spolaor, Q. Yan, C. Zhao, and B. Yang, “DART: Detecting unseen malware variants using adaptation regularization transfer learning,” 2019 IEEE International Conference on Communications (ICC), pp.1–6, 2019.

[3] M. Zolanvari, M.A. Teixeira, and R. Jain, “Effect of imbalanced datasets on security of industrial iot using machine learning,” 2018 IEEE International Conference on Intelligence and Security Informatics (ISI), pp.112–117, 2018.

[4] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, and F. Ahmad, “Network intrusion detection system: A systematic study of machine learning and deep learning approaches,” Transactions on Emerging Telecommunications Technologies, vol.32, no.1, Jan. 2021.

[5] N. Shone, T.N. Ngoc, V.D. Phai, and Q. Shi, “A deep learning approach to network intrusion detection,” IEEE Transactions on Emerging Topics in Computational Intelligence, vol.2, no.1, pp.41–50, 2018.

[6] B. Yan and G. Han, “Effective feature extraction via stacked sparse autoencoder to improve intrusion detection system,” IEEE Access, vol.6, pp.41238–41248, 2018.

[7] M. Al-Qatf, Y. Lasheng, M. Al-Habib, and K. Al-Sabahi, “Deep learning approach combining sparse autoencoder with svm for network intrusion detection,” IEEE Access, vol.6, pp.52843–52856, 2018.

[8] G. Andresini, A. Appice, N.D. Mauro, C. Loglisci, and D. Malerba, “Multi-channel deep feature learning for intrusion detection,” IEEE Access, vol.8, pp.53346–53359, 2020.

[9] F.A. Khan, A. Gumaei, A. Derhab, and A. Hussain, “A novel two-stage deep learning model for efficient network intrusion detection,” IEEE Access, vol.7, pp.30373–30385, 2019.

[10] A. Verma and V. Ranga, “Statistical analysis of CIDDS-001 dataset for network intrusion detection systems using distance-based machine learning,” Procedia Computer Science, vol.125, pp.709–716, Dec. 2017.

[11] R. Abdulhammed, M. Faezipour, A. Abuzneid, and A. AbuMallouh, “Deep and machine learning approaches for anomaly-based intrusion detection of imbalanced network traffic,” IEEE Sensors Letters, vol.3, no.1, pp.1–4, 2019.

[12] C.F.T. Pontes, M.M.C. deSouza, J.J.C. Gondim, M. Bishop, and M.A. Marotta, “A new method for flow-based network intrusion detection using the inverse potts model,” IEEE Transactions on Network and Service Management, vol.18, no.2, pp.1125–1136, 2021.

[13] M. Ring, S. Wunderlich, D. Gruedl, D. Landes, and A. Hotho, “Flow-based benchmark data sets for intrusion detection,” Proceedings of the 16th European Conference on Cyber Warfare and Security (ECCWS), pp.361–369, ACPI, 2017.

[14] M. Ring, S. Wunderlich, D. Gruedl, D. Landes, and A. Hotho, “Creation of flow-based data sets for intrusion detection,” Information Warfare, vol.16, 2017.

[15] L.G. Nguyen and K. Watabe, “Flow-based network intrusion detection based on bert masked language model,” Proceedings of the 3rd International CoNEXT Student Workshop, p.7–8, CoNEXT-SW ’22, Association for Computing Machinery, New York, NY, USA, 2022. https://doi.org/10.1145/3565477.3569152