同一環境下における機械学習アルゴリズムの侵入検知精度比較評価

田河　雅輝[1,a)]　山森　一人[2,b)]　齊藤　燎[3,c)]

概要：NIDS（Network Intrusion Detection System）で用いられる機械学習アルゴリズムの性能を同一環境下で比較評価する．計 7 種の機械学習アルゴリズムを取り上げ，同一環境下で，学習サンプル数が十分な場合と乏しい場合の二つに加えて，学習サンプル中の攻撃通信と通常通信の割合が変化した時の攻撃検出精度についても評価する．実験の結果，Catboost の精度が最も優れており，大量のサンプル数を利用できる場合は LightGBM が，少数サンプルしか利用できない場合は SAINT+PLE が良好な精度を示した．

キーワード：機械学習，分類学習，システムセキュリティ，決定木，深層学習

Comparative of machine learning algorithms for NIDS under the same environment

Abstract: This study compares the performance of machine learning algorithms for NIDS (Network Intrusion Detection System) in the same environment. We pick up 7 machine learning algorithms in three cases. The first is a rich training sample case, the second is a less training sample case. The last is changing the ratio of attack/normal ratio in the training samples. Experiments showed that CatBoost performed the best accuracy, LightGBM and SAINT+PLE for rich and less training samples, respectively.

Keywords: machine learning, classification learning, system security, decision tree, deep learning

1. はじめに

5G などの高速ネットワーク技術の普及や，新型コロナウイルス流行時の遠隔勤務の一般化などにより，ネットワークトラフィックが増加している．ネットワークトラフィックの増加に伴い，サイバー攻撃関連の通信も増加する一方である．総務省 [1] によると，2021 年にはサイバー攻撃関連の通信は 5,180 億パケットにまで達している．コンピュータの安全性を確保するためには，サイバー攻撃関の連通信を検知することが重要な課題になる．大量かつ短期間で変化するサイバー攻撃に対応するために，機械学習を用いて NIDS（Network Intrusion Detection System）の性能を向上させようとする研究 [2-5] が行われている．これらの研究は独自の環境に基づいており，同じ条件での比較は十分ではない．本研究では，NIDS への適用を前提に，データセットやそれに含まれる攻撃サンプルの割合などの条件を揃えて，同一環境下での機械学習アルゴリズムの比較調査を行うことを目的とする．

本研究では，GBDT（Gradient-Boosting-Decision-Tree）系のアルゴリズムから 3 種類，深層学習系アルゴリズムからから 2 種類を採り上げ，同一環境下で評価する．GBDT アルゴリズムからは XGBoost[6], LightGBM[7], CatBoost[8] を，深層学習系アルゴリズムからは SAINT（Self Attention and INtersample attention Transformer）[9]，FT-Transformer（Feature Tokenizer-Transformer）[10] を選定している．さらに数値特徴量のベクトル化に PLE（Piece-

[1] 宮崎大学 工学研究科
Graduate School of Engineering,University of Miyazaki, Japan
[2] 宮崎大学工学教育研究部
Faculty of Engineering, University of Miyazaki, Japan
[3] 宮崎大学 農学工学総合研究科
Interdisciplinary Graduate school of Agriculture and Engineering, University of Miyazaki, Japan
[a)] hm17027@student.miyazaki-u.ac.jp
[b)] yamamori@cs.miyazaki-u.ac.jp
[c)] saitou@taurus.cs.miyazaki-u.ac.jp

wise Linear Encoding) [11] を組み合わせた SAINT+PLE, FT-Transformer+PLE も比較対象に含めて，計 7 種の機械学習アルゴリズムで比較を行う．

比較実験では，学習サンプル数が十分な場合と乏しい場合の二つに加えて，学習サンプルの攻撃通信と通常通信の割合が変化した時の検出精度についても評価する．

2. 比較対象の機械学習アルゴリズム

2.1 XGBoost

XGBoost[6] は，0 の要素が多い疎なデータや，欠損値のあるデータに対しても考慮したアルゴリズムである．具体的には，分岐の際に 0 か欠損値を持つサンプルは，予め決まったノードに振り分けられるように設定することで，これらのサンプルに対処する．XGboost は過学習を防ぐために，正則化項に加えて Column Subsampling と Shrinkage[12] 二つの手法を用いる．

Column Subsampling は学習中に入力サンプルのすべての特徴量を用いず，ランダムに採用する手法である．一方，すべてのサンプルからランダムに選択する手法を Row Subsampling と言い，併用が可能である．特徴量やサンプルをランダムに選択することで，学習サンプルに対する過学習を防ぐ効果がある．

XGBoost の予測値 $\hat{y}_i^t$ を求める際に，各弱学習器の出力 $f_j(\boldsymbol{x}_i)$ にハイパーパラメータ $\eta(0 < \eta < 1)$ を乗じて損失関数の値を小さくし，結果として目的関数の値も小さくして各弱学習器の影響を抑える，Shrinkage と呼ばれる手法を用いることがある．Shrinkage を踏まえた予測値 $\hat{y}_i^t$ を式 (1) に表す．

$$\hat{y}_i^t = \eta \sum_{j=1}^{t} f_j(\boldsymbol{x}_i). \tag{1}$$

η が小さ過ぎると学習の進みが遅くなり，学習にかかる時間が増加することに注意が必要である．

2.2 LightGBM

LightGBM[7] は，学習速度が極めて速いことが特徴の GBDT アルゴリズムである．高速な学習を実現するために，GOSS(Gradient-based One-Side Sampling) と EFB(Exclusive Feature Bundling) の二つの手法が用いられている．

GOSS はパラメータ変動に対する勾配が大きい上位 $a\%$ のサンプルを，残ったサンプルから $b\%$ をランダムに選択し学習を行う手法であり，すべての学習サンプルを学習しない分，学習速度が上昇する．

EFB は二つの特徴量を特徴量を一つにまとめる手法であり，特徴量が減る分しきい値の候補が減り，その分学習速度が上昇する．

2.3 CatBoost

CatBoost[8] は，非数値特徴量であるカテゴリ特徴量の処理に優れたアルゴリズムであり，カテゴリ特徴量をラベル $\boldsymbol{y}$ の期待値に変換する TS(Target statistics) を改良した Ordered TS を用いる．一般に，TS を使うと過学習を起こしやすいので，過学習を防ぐために Ordered TS が用いられる．

サンプル数 N の k 番目のサンプルの，あるカテゴリ特徴量を c_k，ラベルを y_k とした時，TS による c_k の変換結果 $\hat{c}_k(1 \leq k \leq N)$ を式 (2) に示す．

$$\hat{c}_k = \frac{\sum_{i=1}^{N} \mathbb{1}\{c_k = c_i\} y_i}{\sum_{i=1}^{N} \mathbb{1}\{c_k = c_i\} + \varepsilon.} \tag{2}$$

ここで，ε は分母が 0 にならないようにするための微小な数値であり，$\mathbb{1}\{c_k = c_i\}$ は $c_k = c_i$ のとき 1 をとる関数である．Ordered TS での，カテゴリ特徴量 c_k の変換結果 $\hat{c}_{\sigma_k}$ は式 (3) となる．

$$\hat{c}_{\sigma k} = \frac{\sum_{i=1}^{k-1} \mathbb{1}\{c_k = c_i\} y_i}{\sum_{i=1}^{k-1} \mathbb{1}\{c_k = c_i\} + \varepsilon.} \tag{3}$$

TS では最初のサンプルから最後のサンプルまでの，あるカテゴリ特徴量の出現回数を用いるのに対して，OrderedTS は変換対象の一つ前の $(k-1)$ 番目までの出現回数を用いる点に違いがある．TS の場合はカテゴリ特徴量が同じなら同じ数値となるが，OrderedTS の場合は同じカテゴリ特徴量でもその期待値はサンプルの出現位置により異なる値になる．

2.4 FT-Transformer

FT-Transformer[10] は，自然言語処理に向けて開発された深層学習系アルゴリズムである BERT[13] と Transformer[14] の構造を参考に開発された，表形式データ用の深層学習系アルゴリズムであり，1 サンプルの特徴量間の類似度を考慮している．どのようなサンプルを学習するか予習させる事前学習は行わない．

2.5 SAINT

SAINT[9] は表形式データ用深層学習系アルゴリズムの一つで，FT-Transformer 同様に BERT と Transformer をベースにしている．SAINT は 1 サンプルの特徴量間の類似度だけでなく，サンプル間の類似度も考慮している．また，事前学習を採り入れ，事前学習後に問題タスクへの調整 (Fine-Tuning) を行うアルゴリズムとなっている．

2.6 PLE

PLE(Piecewise Linear Encoding) は数値特徴量を one-hot 表現に類似した形式にエンコードする手法である．one-hot 表現とは，一つの成分が 1 で残りすべてが 0 になる表現であり，カテゴリ特徴によく用いられる．例え

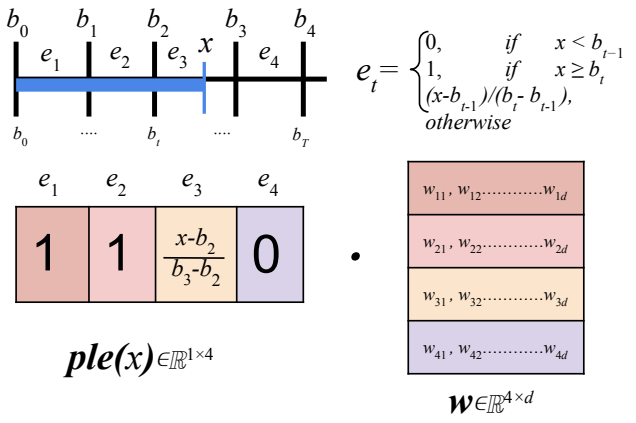

図 1 PLE によるベクトル化の例.

ば，ある特徴量に 4 つのカテゴリ A，B，C，D があり，これらを one-hot 表現にすると，A:(1,0,0,0)， B:(0,1,0,0)，C:(0,0,1,0)， D:(0,0,0,1) のようになる．PLE では予め二つの値の区間 $B_t = [b_{t-1}, b_t)$ を $t(1 \leq t \leq T)$ 個用意し，数値特徴量 x が b_t 以上の時 $e_t = 1$，b_{t-1} 未満の時 $e_t = 0$，$b_{t-1} \leq x \leq b_t$ の時 $e_t = (x - b_{t-1})/(b_t - b_{t-1})$ にエンコードする．

PLE によるベクトル化の例を図 1 に示す．図 1 の場合は区間が 4 つあるので $\boldsymbol{PLE}(x)$ は 4 次元のベクトルになるが，$\boldsymbol{PLE}(x)$ は区間数 T に従い次元数が変わるので，一般に T 次元のベクトルになる．x を PLE によりベクトル化した $\boldsymbol{v}$ は式 (4) で定義される．

$$\boldsymbol{v} \quad = \quad \boldsymbol{PLE}(x) \cdot \boldsymbol{W} + \boldsymbol{bias}. \tag{4}$$

ここで，$\boldsymbol{W} \in \mathbb{R}^{T\times d}$ の重み，$\boldsymbol{bias} \in \mathbb{R}^d$ はバイアスである．

二つの値の区間 $B_t = [b_{t-1}, b_t)$ の決め方について説明する．区間 $B_t = [b_{t-1}, b_t)$ はサンプルの各特徴を昇順に並び替え，データを T 分割した際の区切りである分位数を b_t に用いる．本研究で用いる NSL-KDD[15] データセットは各特徴量に 0 が多いので，分位数が重複する．b_t が重複した場合は，重複を一つにまとめて昇順に並び替え，分位数をとる．もし重複を一つにまとめた結果データ数が T 未満になり，T 個の分位数が取れない場合は，最小値と最大値の間から等間隔で t 分割する．本研究では，$T = 4$ で比較実験を行う．

3. 比較実験

3.1 実験環境

NSL-KDD データセットを用い，CatBoost(CatB)，LightGBM(LGB)，XGBoost(XGB)，SAINT，SAINT+PLE(SAINT+)，FT-Transformer(FTT)，FTT+PLE(FTT+) の計七種について比較評価を行う．各機械学習アルゴリズムにおける前処理を表 1 に示す．数値特徴量の前処理は，PLE を用いない場合には

表 1 数値特徴とカテゴリ特徴に対する前処理.

	数値特徴	カテゴリ特徴
CatB	無し	Label-Encode
LGB	無し	Label-Encode
XGB	無し	One-Hot-Encode
SAINT	Min-Max-Scaler	Label-Encode
SAINT+	無し	Label-Encode
FTT	Min-Max-Scaler	Label-Encode
FTT+	無し	Label-Encode

特徴量の最大値を 1，最小値を 0 になるように変換する Min-Max-Scaler で正規化を行い，他の場合では前処理無しで学習を行う．カテゴリ特徴量には Label-Encode で各カテゴリを整数に変換する．Label-Encode は，例えば ‘black’，‘white’，‘red’ といったカテゴリがある場合，‘black’: 0，‘white’: 1，‘red’: 2 に変換するエンコードである．カテゴリ特徴量を扱えない XGBoost は one-hot 表現で整数化する．

GBDT 系アルゴリズムのパラメータはライブラリのデフォルトを用い，深層学習系アルゴリズムのパラメータも原著論文の推奨パラメータで使用する．深層学習系アルゴリズムのエポック数は事前学習，Fine-Tuning 共に 100 回に固定する．対象とする問題は，通信を通常と攻撃の二つに分類するので二値分類として扱い，分類のしきい値は 0.5 とする．

一般に，二値分類問題の評価には図 2 に示す混同行列を用いる．本研究では NSL-KDD のサンプルを攻撃通信，通常通信に分けることになり，行方向に正解ラベル，列方向に予測結果を並べた行列になる．各アルゴリズムの評価に用いる正解率と F 値の定義は以下の通りである．

$$\begin{aligned} Acc \quad &= \quad \frac{TP+TN}{TP+TN+FP+FN} \times 100, \\ F-measure \quad &= \quad \frac{2TP}{2TP+FP+FN} \times 100. \end{aligned} \tag{5}$$

上式で TP，TN，FP，FN は図 2 に示した通りである．

正解率は全サンプル中でラベルと予測結果が一致する割合を表す．F 値は，攻撃のラベルが付与されたサンプルのうち，攻撃と予測されたものの割合を指す再現率 $\frac{TP}{TP+FN} \times 100$ と，攻撃と予測したサンプルのうち，実際に攻撃のラベルを持つサンプルの割合を指す適合率 $\frac{TP}{TP+FP} \times 100$ のバランスを評価できる指標で，両方がバランス良く高いと F 値も高くなる．

AUROC は，通常通信と攻撃通信を分類する際のしきい値 θ を 0 から 1 まで変動させて，$TPR = \frac{TP}{TP+FN}$ を縦軸に，$FPR = \frac{FP}{FP+TN}$ を横軸にプロットした曲線と，$FPR = 1$ の直線で囲まれる部分の面積である．面積を最大とする θ の時が最も誤りなく分離できている状態を指すので，AUROC は 1 に近いほど良い．図 3 に AUROC の概念を示す．

評価では，正解率と F 値，AUROC の順位に基づいて評価点を与える．評価点の付け方は，各実験でのアルゴリズ

		予測	
		通常	攻撃
真の値	通常	TN (True Negative)	FP (False Positive)
	攻撃	FN (False Negative)	TP (True Positive)

図 2 混同行列の例.

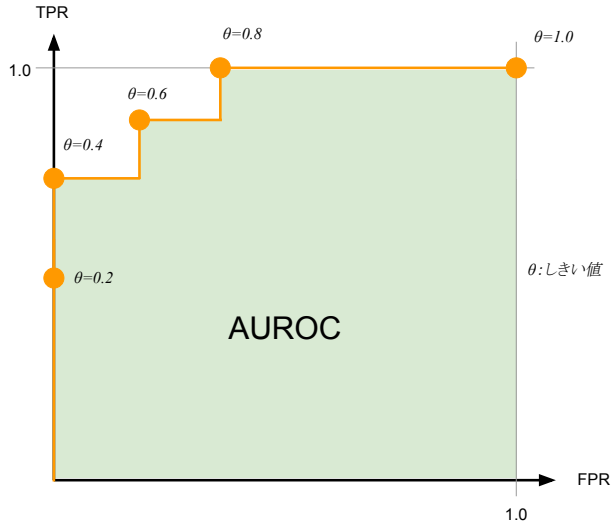

図 3 AUROC の定義.

ムごとに，(アルゴリズムの数 (7) - 順位) とする．例えば，1 位のアルゴリズムには (7 – 1) で 6 点，2 位のアルゴリズムには (7 – 2) で 5 点と，順位が下がるにつれて 1 点づつ減少させる．最下位のアルゴリズムは 0 点になる．

正解率と F 値，AUROC の総合評価では，各評価点を合算した上位 3 つのアルゴリズムに◎，最下位に △，それ以外には ◯ で表す．

各アルゴリズムは python3.8.8 で記述し，Microsoft Windows 10 上に実装した．実験に用いたハードウェア環境を表 2 に，実装に使用した機械学習アルゴリズムのソフトウェアパッケージを表 3 にまとめる．

3.2 実験 1: 大量サンプル数での比較実験

サンプル数が大量に用意できる場合について評価する．評価では分割交差検証を行う．分割交差検証とは，学習データセットを複数のサブセットに分割し，サブセット内の一つを検証データセットとして用い，他を学習データセットとして利用する手法である．分割数は 6 で学習用データセット $\frac{5}{6}$，個数でいうと 104,977 個を学習に使い，テストデータは用意されている 22,544 個のをそのまま用いる．サブセットの組み合わせを替えつつ 6 回の学習，テストを行い，結果をまとめる．

表 2 使用したハードウェアと言語.

CPU	Intel(R) Core(TM) i7-7700 CPU
メモリ	16.0 GB
GPGPU	NVIDIA Quadro M4000 8.0GB
OS	Windows 10 Pro, build 19044.2486
使用言語	python 3.8.8
開発環境	VScode 1.74.3

表 3 使用したパッケージとバージョン.

	package
CatB	catboost 1.0.6
LGB	lightgbm 3.3.2
XGB	xgboost 1.6.1
SAINT	pytorch 1.12.1 で自作
SAINT+	pytorch 1.12.1 で自作
FTT	pytorch 1.12.1 で自作
FTT+	pytorch 1.12.1 で自作

実験により得た各評価指標の平均を図 4 に示す．正解率 (Accuracy) は上位から FTT，LGB，CatB の順になり，F 値 (以下，図中では F-measure で表す) は FTT，LGB，SAINT，AUROC は CatB，XGB，SAINT+の順となり，総合評価は CatB が 1 位，LGB が 2 位，同点で XGB と FTT が 3 位となった．

この結果から，サンプル数が大量にある場合では CatB，LGB，XGB，FTT が比較的良い精度を示すことが分かった．FTT が高い性能を示している一方で，FTT+の性能が振るわないのは，PLE により特徴量の次元が増えた分，予測を阻害する余計な情報までも学習したためだと考えられる．Leo ら [16] は，「表形式データには無駄な特徴量が含まれていた場合，深層学習のモデルは役に立たない特徴量も無視できずに使い学習を行うため，従来の木構造の機械学習アルゴリズムに敵わない」と指摘している．

LGB の正解率と F 値が比較的に高かったことについて考える．正解率と F 値の定義から，正解率と F 値が大きくなるのは分子が大きいか分母が小さいかのどちらかである．分母は予測の度に変わることはないので，正解率と F 値の両方の分子に現れる TP が高いと推測できる．TP が大きくなる理由については，今回の実験では明らかにするまでには至らず今後の調査が必要である．

3.3 実験 2: 少数サンプル数での比較実験

サンプル数を減らした時にどの程度精度を維持できるか，サンプル数の減少に対する頑健性を評価する．サンプル数は少ない順に 10，100，1,000，10,000 個とし，学習と

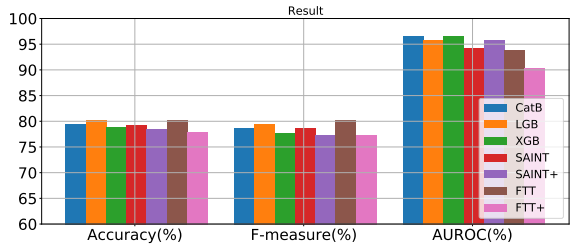

図 4 サンプルが大量に準備できる場合の各評価指標． 正解率:Accuracy，F 値:F-measure，AUROC:AUROC，y 軸:%．

表 4 サンプルが大量に準備できた場合の各アルゴリズムの評価点と総合評価．

	正解率	F 値	AUROC	総合評価
CatB	4	3	6	◎
LGB	5	5	3	◎
XGB	2	2	5	◎
SAINT	3	4	2	○
SAINT+	1	1	4	○
FTT	6	6	1	◎
FTT+	0	0	0	△

予測を 10 回行い，各指標の平均を求めて，順位をもとに評価点を与える．入力するサンプルは試行回ごとにランダムで取得するが，同一試行回で各アルゴリズムに入力するサンプルは同じものにする．

正解率の変化を示す図 5 では，サンプル数 10 での正解率は CatB，XGB，SAINT，SAINT+が比較的に高く，サンプル数 100 で LGB，FTT，FTT+の正解率が上昇するが，FTT と FTT+はやや低い．十分なサンプル数がある場合と比べて FTT の正解率が低いのは，少ないサンプルに対する過学習が原因だと考えられる．実際に FTT の学習データ，検証データ，テストデータの正解率を調べると，学習データで 77.56%，検証データで 53.46%，テストデータで 43.08%を示し過学習の状態であることが分かった．

F 値の変化を示す図 6 では，サンプル数 10 での F 値は正解率と同様に CatB，XGB，SAINT，SAINT+が比較的に高く，サンプル数 100 で LGB，FTT，FTT+の正解率が上昇するが，FTT と FTT+はやや低い．FTT の F 値が十分なサンプル数がある場合と比べて低いのは，正解率と同様に少ないサンプル数による過学習が原因と考えられる．サンプル数 1,000 での F 値では各アルゴリズムで大きな差はなく，サンプル数 10,000 ですべてのアルゴリズムが同等となった．

AUROC の変化を示す図 7 では，サンプル数 10 での AUROC は CatB，SAINT，SAINT+，FTT，FTT+が比較的高い．サンプル 1,000 ではどのアルゴリズムでも大きな差はなく，サンプル 10,000 では SAINT，FTT，FTT+が低くなった．十分なサンプル数がある時も SAINT，FTT，FTT+の AUROC は低いことを考えると，SAINT，FTT，

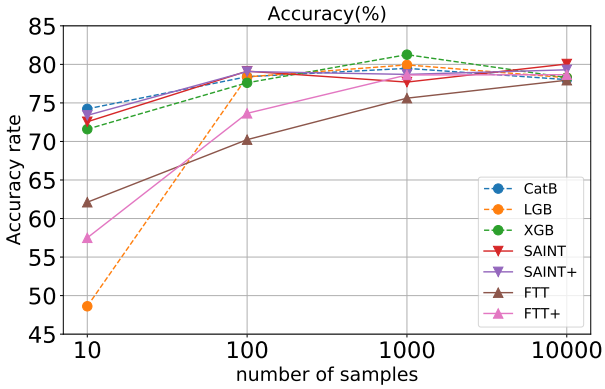

図 5 サンプル数を変化させたときの正解率の推移．x 軸: 学習サンプル数， y 軸: 正解率 (%)．

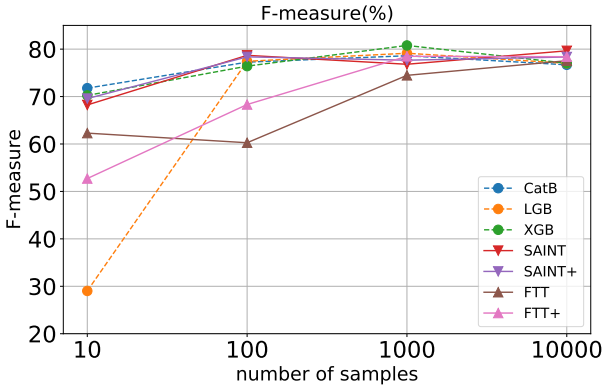

図 6 サンプル数を変化させたときの F 値の推移．x 軸: 学習サンプル数， y 軸:F 値 (%)．

FTT+はサンプル数を増やしても AUROC が上がりづらい性質が見てとれる．

各指標の評価点をサンプル数ごとに与え合計した結果を表 5 に示す．表 5 を見ると，十分なサンプル数がある場合と異なり，SAINT，SAINT+の精度が比較的高い結果となった，その一方で，同じ深層学習系アルゴリズムの FTT と FTT+の精度が低い．SAINT と FTT は構造が似ており，異なる点は事前学習と Intersample-Attention の有無である．このことから，Intersample-Attention，もしくは事前学習が頑健性の向上に効果がある考えられる．CatB が事前学習無しでも SAINT と SAINT+に近い精度を示しているのは，Ordered TS がデータ拡張に近い効果を発揮し，それが少数サンプルに対する頑健性を向上させたと考えられる．以上をまとめると，少数サンプルに対する頑健性が高いのは，SAINT+と CatB，SAINT と結論付けられる．

3.4 実験 3: 攻撃サンプルの割合を変化させた時の比較実験

サンプル数を 10,000 に固定し，攻撃サンプルの割合を 10%から 90%まで，10%づつ変化させた時の性能を評価する．他の実験同様に，試行回ごとにサンプルはランダム抽

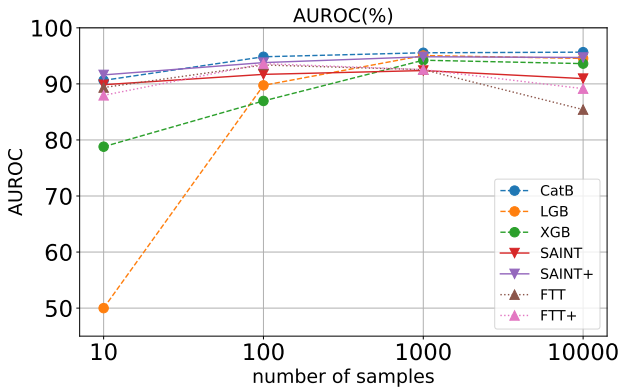

図 7 サンプル数を変化させたときの AUROC の推移．x 軸: 学習サンプル数，y 軸:AUROC(%).

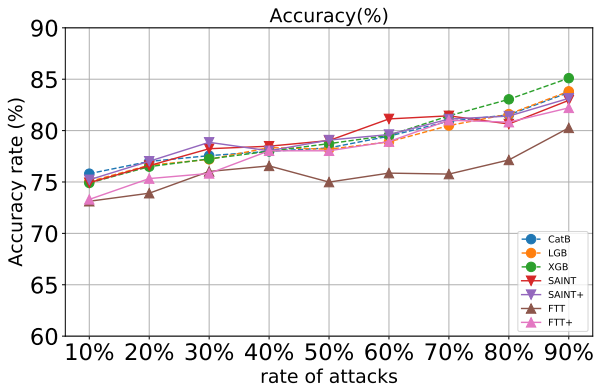

図 8 攻撃サンプルの割合を変化させた時の正解率の推移．x 軸: 攻撃サンプルの割合，y 軸: 正解率 (%).

表 5 少数サンプル数での比較実験による各アルゴリズムの評価点の合計と総合評価.

	正解率	F 値	AUROC	総合評価
CatB	15	13	23	◎
LGB	10	10	10	○
XGB	14	15	7	○
SAINT	17	16	8	◎
SAINT+	18	15	20	◎
FTT	2	5	7	△
FTT+	8	10	9	○

出するが，各アルゴリズムに入力するサンプルは各試行回で同じものにする．サンプル数を 10,000 に固定したのは，少数サンプルの場合の比較結果から，すべてのアルゴリズムの分類精度が同等である方が，攻撃サンプルの割合の変化による指標の変動が現れやすいと判断したためである．

正解率の変化を示す図 8 では，どのアルゴリズムでも攻撃サンプルの割合が増えるほど正解率は上昇しており，多様な攻撃サンプルを学習すれば正解率が上昇すること示している．攻撃サンプルの割合 20%から 70%までは SAINT と SAINT+が上位で，80%から GBDT アルゴリズムがそれらをやや上まわる．少数サンプルでの頑健性が高い SAINT，SAINT+と CatB が攻撃サンプルの割合 10%から上位に位置しており，攻撃サンプルの割合の変化に対する頑健性もあると考えられる．

F 値の変化を示す図 9 では，どのアルゴリズムでも F 値は最低でも 70%以上であることから，攻撃サンプル割合が低い状態で学習しても，攻撃を見落とすことは少ないと言える．正解率と同様に攻撃サンプルの割合が増えるほど F 値は上昇し，攻撃サンプルの割合 20%から 70%までは SAINT と SAINT+が上位で，80%から GBDT アルゴリズムが上まわる．SAINT，SAINT+と CatB が攻撃サンプルの割合 10%から高い F 値を示しており，F 値から見てもこれらは攻撃サンプルの割合の変化に対する頑健性をもつと考えられる．

AUROC の変化を示す図 10 から，GBDT アルゴリズム

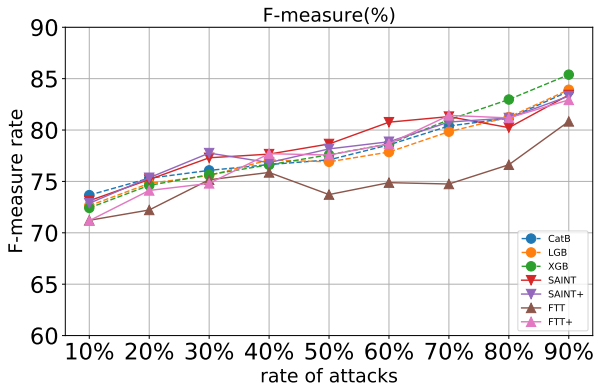

図 9 攻撃サンプルの割合を変化させた時の F 値の推移．x 軸: 攻撃サンプルの割合，y 軸:F 値 (%).

と SAINT+は攻撃の割合が変わっても大きな変化がないことが分かる．一方で，SAINT と FTT，FTT+がサンプル数の大小によらず AUROC が低かったのと同様，ここでも AUROC は低い．AUROC の精度の変化は，サンプル数やサンプル内の内訳は大きく影響しない分，アルゴリズムに依存していると考えられる．

各指標の評価点を攻撃サンプルの割合ごとに付与し合計した結果を表 6 にまとめる．正解率は高い順に SAINT+，SAINT，XGB となり，F 値は SAINT，SAINT+，AUROC は CatB，LGB，SAINT+となり，総合評価では CatB，SAINT+，LGB の順になった．AUROC では攻撃サンプルの割合の変化による影響はあまり見られないが，正解率と F 値は，攻撃サンプルの割合が増えれば増えるほど，すべてのアルゴリズムで上昇している．一般に，機械学習アルゴリズムでは分類したいクラスに属するサンプル数はそれぞれ均等にすることが推奨される．本研究の場合，攻撃という 1 つのラベルに多種多様な攻撃サンプルが含まれており，これらを均等に学習するためには多くの攻撃サンプルが必要になる．そのため，攻撃サンプルの割合が高い方が，正解率と F 値が上昇したと考えられる．

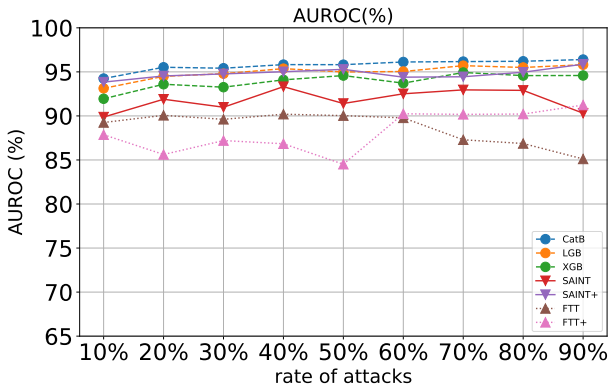

図 10 攻撃サンプルの割合を変化させた時の AUROC の推移．x 軸: 攻撃サンプルの割合，y 軸:AUROC(%).

表 6 攻撃サンプルの割合を変えた時の各アルゴリズムの評価点の合計と総合評価.

	正解率	F 値	AUROC	総合評価
CatB	32	29	54	◎
LGB	29	25	41	◎
XGB	34	31	28	○
SAINT	37	40	17	○
SAINT+	42	37	39	◎
FTT	1	2	5	△
FTT+	14	25	5	○

表 7 3 つの比較実験の総合評価.

	実験 1	実験 2	実験 3
CatB	◎	◎	◎
LGB	◎	○	◎
XGB	◎	○	○
SAINT	○	◎	○
SAINT+	○	◎	◎
FTT	◎	△	△
FTT+	△	○	○

3.5 実験のまとめ

3 つの比較実験の総合評価をまとめた結果を表 7 に示す．表中の実験 1 はサンプルが大量に準備できた場合，実験 2 はサンプル数を変化させた場合，実験 3 は攻撃サンプルの割合を変えた時の各アルゴリズムの総合評価をそれぞれ示す．

表 7 から，CatBoost がすべての実験で良好な成績を示しており，次に LightGBM，SAINT+の順に性能が良いことが分かる．NSL-KDD データセットは，テストデータセットのうち，学習データにはない種類の攻撃サンプルが半分近くを占めており，学習データに対して高精度な結果を示しても，テストデータの予測では誤ることが多い．CatBoost がどの実験でも比較的高性能なのは，サンプル数に大きく左右されない過学習対策である Ordered TS が大きな要因だと考えられる．

3.6 終わりに

5G などの高速ネットワーク技術の普及や，新型コロナウイルス流行時の遠隔勤務の一般化などにより，ネットワークトラフィックが増加している．ネットワークトラフィックの増加に伴い，サイバー攻撃関連の通信も増加している．大量かつ短期間で変化するサイバー攻撃に対応するために，機械学習を用いて NIDS の性能を向上させようとする研究が行われているが，これらの研究は独自の環境に基づいており，同じ条件での比較は十分ではない．本研究では，データセットやそれに含まれる攻撃サンプルの割合などの条件を揃えて，同一環境下での機械学習アルゴリズムの比較調査を行うことを目的とした．本研究では，NIDS を対象とした機械学習アルゴリズムの性能を，同一環境下で，どのような状況で高精度な分類結果を示すのか調査を行った．

比較対象の機械学習アルゴリズムは GBDT 系アルゴリズムから XGBoost，LightGBM，CatBoost の 3 種，深層学習系アルゴリズムから FT-Transformer，SAINT の 2 種，それらに PLE を加えた FT-Transformer+PLE，SAINT+PLE の二つ，計 7 種を採り上げた．

サンプル数が多い場合，正解率 (Accuracy) について上位から FT-Transformer，LightGBM，CatBoost の順になり，正解率は 80% 近くまでに達した．F 値 (F-measure) は FT-Transformer, LihghtGBM, SAINT の順となり，AUROC は CatBoost，XGBoost，SAINT +PLE の順となり AUROC は 95% を超えることが分かった．総合評価では CatBoost，LightGBM，同点で XGBoost と FT-Transformer の順になった．

サンプル数が少ない場合，サンプル数 10 での正解率は CatB，XGB，SAINT，SAINT +PLE が 70%を超えた．サンプル数 100 での LGB，FTT，FTT+の正解率も 70%以上まで上昇する．サンプル数 10 での F 値は正解率と同様に CatB，XGB，SAINT，SAINT+が 70%程度，サンプル数 1,000 では，各アルゴリズムで大きな差はなく，サンプル数 10,000 ですべてのアルゴリズムが同等の値を示した．

サンプル数 10 での AUROC は CatB, SAINT, SAINT+, FTT，FTT+が 85%を超え，サンプル数 1,000 ではどのアルゴリズムでも大きな差はなく 90%を超えた．正解率，F 値，AUROC をまとめた総合評価では SAINT+PLE, CatBoost，SAINT の順になった．

攻撃サンプルの割合を変えての比較実験では，どのアルゴリズムでも攻撃の割合が増えるほど正解率が上昇しており，豊富かつ多様な攻撃サンプルを学習すれば正解率が上昇すること示した．攻撃サンプルの割合が 20%から 70%までは，SAINT と SAINT+が上位で，80%から GBDT アルゴリズムがやや上回った．F 値も同様に攻撃サンプルの割合が 80%以上から GBDT アルゴリズムが優れていた．AUROC では，CatBoost が常に最上位に位置し，続

いて LGB と SAINT+が並ぶ形が多かった．正解率，F 値，AUROC をまとめた総合評価では CatBoost，SAINT+，LGB の順になった．

実験の結果，CatBoost が常に正解率は 73%以上，F 値は 70%以上，AUROC は 90%以上に達し，他のアルゴリズムよりも高精度となった．LightGBM は最低でもサンプルは 100 個必要となることが分かった．SAINT+PLE は比較的に高性能だが，大量サンプルでの学習では GBDT アルゴリズムよりも劣ることが分かった．どの機械学習アルゴリズムでも，攻撃の割合が増えるほど正解率は上がることを示した．まとめると，どの場合でも Catboost の精度は上位になり，大量のサンプル数を利用できる場合は LightGBM が，少数サンプルしか利用出来ない場合は SAINT+PLE が良いとの結果を得た．

今後課題として，NSL-KDD データセット以外での比較と，他の深層学習系アルゴリズムを加えての比較，大量のデータが利用できる場合に LightGBM の TP が高い原因の解明が挙げられる．

謝辞 本研究は JSPS 科研費 JP19K12139 の助成により行われたものです．

参考文献

[1] 総務省：令和 4 年版 情報処理白書 第 2 部，入手先 ⟨https://www.soumu.go.jp/johotsusintokei/whitepaper/ja/r04/pdf/01honpen.pdf⟩ (2023.1.20).

[2] Mohd R.Z., Zuhairi M.F., Shadil A.Z. and Dao H.: “Anomaly-based nids: A review of machine learning methods on malware detection”, 2016 International Conference on Information and Communication Technology (ICICTM), pp. 266–270, (2016).

[3] Jin D., Lu Y., Qin J., Cheng Z. and Mao Z.: “KC-IDS: Multi-layer intrusion detection system”, *2020 International Conference on High Performance Big Data and Intelligent Systems (HPBD&IS)*, pp. 1–5, (2020).

[4] Tang C., Luktarhan N. and Zhao Y.: “An efficient intrusion detection method based on LightGBM and autoencoder”, *Symmetry*, Vol. 12, No. 9, pp. 1458–1474, (2020).

[5] Dhaliwal S. S., Nahid A. and Abbas R.: “Effective intrusion detection system using XGBoost”, *Information*, Vol. 9, No. 7, pp. 149–173, (2018).

[6] Chen T. and Guestrin C.: “Xgboost: A scalable tree boosting system”, *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, (2016).

[7] Ke G., Meng Q., Finley T., Wang T., Chen W., Ma W., Ye Q. and Liu T.: “Lightgbm: A highly efficient gradient boosting decision tree”, *Advances in neural information processing systems*, Vol. 30, pp. 3149–3157, (2017).

[8] Dorogush A.V., Ershov V. and Gulin A.: “CatBoost: gradient boosting with categorical features support”, *arXiv preprint arXiv:1810.11363*, (2018).

[9] Somepalli G., Goldblum M., Schwarzschild A., Bruss C.B.,Goldstein T.: “Saint: Improved neural networks for tabular data via row attention and contrastive pretraining”, *arXiv preprint arXiv:2106.01342*, (2021).

[10] Gorishniy Y., Rubachev I., Khrulkov V. and Babenko A.: “Revisiting deep learning models for tabular data”, *Advances in Neural Information Processing Systems*, Vol. 34, pp. 18932–18943, (2021).

[11] Gorishniy Y., Rubachev I. and Babenko A.: “On Embeddings for Numerical Features in Tabular Deep Learning”, *arXiv preprint arXiv:2203.05556v2*, (2022).

[12] Friedman J.H.: “Stochastic gradient boosting”, *Computational statistics & data analysis*, Vol. 38, No. 4, pp. 367–378, (2002).

[13] Devlin J., Chang M., Lee K. and Toutanova K.: “Bert: Pre-training of deep bidirectional transformers for language understanding”, *arXiv preprint arXiv:1810.04805* (2018).

[14] Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A.N., Kaiser Ł. and Polosukhin I.: “Attention is all you need”, *Advances in neural information processing systems*, Vol. 30, pp. 6000–6010, (2017).

[15] Tavallaee M., Bagheri E., Lu W. and Ghorbani A.: “A detailed analysis of the KDD CUP 99 data set”, *2009 IEEE symposium on computational intelligence for security and defense applications*, pp. 1–6, (2009).

[16] Grinsztajn L., Oyallon E. and Varoquaux G.: “Why do tree-based models still outperform deep learning on tabular data?” *arXiv preprint arXiv:2207.08815*, (2022).