Computer Security Symposium 2020
26 - 29 October 2020

ニューラルネットワークを用いたハードウェアトロイ識別に対するデータ拡張と敵対的学習の応用と評価

野澤 康平[1,a)]　披田野 清良[2]　清本 晋作[2]　戸川 望[1]

概要：近年，IC 製品の需要増加により，設計・製造工程の外部委託が増加している．各工程に第三者が関与することで，ハードウェアトロイと呼ばれる悪意ある機能を持つ回路を挿入される脅威が高まっている．ハードウェアトロイ識別の有効性が確認されている対策手法の 1 つに，回路設計情報から抽出した特徴量を利用し，ニューラルネットワークなどの機械学習を用いて識別する手法がある．機械学習による識別は有効である一方，入力に特殊な改変を加えて識別結果を操作してしまう攻撃 (敵対的サンプル攻撃) も存在している．本稿では，敵対的サンプル攻撃に対しても堅牢なハードウェアトロイ識別器を生成するために，データ拡張と敵対的学習と呼ばれる技術をハードウェアトロイ識別にも応用できるかを実証及び検討する．敵対的学習では，訓練データの一部を敵対的サンプルに改変しながら学習を行うことで，敵対的サンプルに耐性のあるモデルを構築する．

キーワード：敵対的サンプル，敵対的学習，ハードウェアトロイ，機械学習，ネットリスト

Evaluation of Data Augmentation and Adversarial Training to Hardware-Trojan Detection Utilizing Neural Networks

Kohei Nozawa[1,a)]　Seira Hidano[2]　Shinsaku Kiyomoto[2]　Nozomu Togawa[1]

Abstract: Recently, outsourcing of IC design and manufacturing steps to third parties is increasing because of great demand of integrated circuits (ICs). At the same time, the threat of injecting a malicious circuit, called a hardware Trojan, by third parties has been increasing. Machine learning is one of the powerful solutions of detecting hardware Trojans. However, weakness of such a machine-learning-based classification method against adversarial examples (AEs) has been reported, which causes incorrect classification by adding perturbation in input samples. This paper proposes a framework applying data augmentation and adversarial training techniques to hardware-Trojan detection at gate-level netlists utilizing neural networks. In adversarial training, we construct robust model against AEs by learning with train data partially replaced with AEs. We, then, demonstrates the effectiveness of these techniques by conducting experiments utilizing Trust-HUB benchmarks.

Keywords: adversarial examples, adversarial training, hardware Trojan, machine learning, netlist

1. はじめに

近年，IoT (Internet of Things) が産業界や一般家庭で普及し，産業ロボットや家電などの多くの製品がインターネットに接続されている．IoT 製品の増加の結果，IC の需要も高まっている．ハードウェアの品質を保ちながら低コストで製造することは設計・製造工程での課題となっている．低コストで効率的にハードウェアを製造する際に用いられるのが，サードパーティへの設計・製造の委託である．

ハードウェア製品を構成する IC が完成するまでの工程

[1] 早稲田大学大学院基幹理工学研究科情報理工・情報通信専攻
Dept. Computer Science and Communications Engineering, Waseda University
[2] 株式会社 KDDI 総合研究所
KDDI Research, Inc.
[a)] kohei.nozawa@togawa.cs.waseda.ac.jp

は，設計工程と製造工程の 2 つに大きく分けられる．IC の設計工程では，ハードウェアベンダから指定された製品仕様を満たすような回路を，ハードウェア記述言語と呼ばれる専用の言語で記述し，設計・構成する．この工程の中で，サードパーティにハードウェアの設計を委託したり，プロセッサや I/O などの汎用的な機能を持つ回路を，IP (Intellectual Property) と呼ばれるパッケージ化されたモジュールとしてサードパーティから購入して製品に組み込んだりすることがある [1]．IC の製造工程では，ファウンドリと呼ばれる製造工場で，設計工程で生成された回路情報をもとに回路が製造される．このように，ハードウェア製品が完成するまでに，多くのサードパーティが関与する．

一方，ハードウェアの設計・製造工程において，ハードウェアに悪意のある機能を挿入される危険性が指摘されている [2], [3]．ハードウェアに挿入された悪意のある機能を持った回路はハードウェアトロイと呼ばれる [4]．ハードウェアトロイは，トリガ回路とペイロード回路の 2 種類の回路から構成される [5]．トリガ回路は，通常の回路から信号をバイパスするなどして，トロイ回路の起動条件を満足するかを監視する回路である [3]．起動条件は，プライマリ入力の値や内部状態などである．ペイロード回路は，トリガ回路がオンになると駆動する，悪意のある機能本体の回路である．悪意のある機能には，外部との通信による情報漏洩，IC の性能低下および機能停止などがある．多くの IoT 機器は，ハードウェアトロイに感染する可能性を否定できない．IoT 機器の普及が進む現在，ハードウェアトロイの効率的な検出は喫緊の課題である．ハードウェアトロイは，挿入される通常の回路と比較してごく小規模な回路であるため，その特徴をいかに見つけるかが重要である．

ここで，回路の設計工程に焦点を当てる．設計工程は IC を製造する工程の最初の段階であり，ここでハードウェアトロイの脅威を除去できると効率が良く理想的である．いま，回路中で使用される素子とその配線を記述した論理レベルの回路設計情報を考える．回路内の配線はネットと呼ばれ，論理レベルで表現された設計情報はネットリストと呼ばれる．回路中で使用される素子は大規模なものでは数百万もの規模になり，大規模なネットリストを精査するのは困難である．論理レベルのネットリストにおけるハードウェアトロイ識別では，大規模なネットリストからいかにハードウェアトロイを見つけ出すかが課題となる．ハードウェアトロイ識別手法には，論理レベルのネットリストにおけるハードウェアトロイ回路の特徴からネットをスコア付けして識別する手法 [6] をはじめ，論理レベルのネットリストから特徴量を抽出し，機械学習により識別する手法 [7] などが提案されている．

機械学習による学習および識別も万全ではない．一つの脅威として，識別器に誤識別を引き起こすような攻撃手法 [8], [9] が提案されている．この攻撃では，学習済みの機械学習モデルを対象として，テストデータに摂動と呼ばれるノイズを加えることで，本来分類されるべきクラスとは全く異なるクラスに分類されるという結果をもたらす．誤識別を引き起こすこのようなテストデータを敵対的サンプル (Adversarial Example, AE) と呼び，本稿ではこれを用いた攻撃を AE 攻撃と呼ぶ．AE 攻撃の手法は多数提案されており，いかに対策を講ずるかが深刻な課題となっている [10]．初期の AE に関する研究は，画像認識を対象としていたが，近年他の対象にも焦点を当てているものが登場している．例えば，物体認識を対象とした AE 攻撃が提案されている [11]．

既存の機械学習によるハードウェアトロイ識別手法は，AE 攻撃などの高度な攻撃を想定していない．それゆえ，AE 攻撃を受ける脅威にさらされている．ハードウェアトロイ識別における AE 攻撃は，識別器を騙すことでハードウェアトロイ回路を正常な回路と誤識別させる可能性がある．攻撃が成功すれば，ハードウェアトロイ識別はより困難になる．先行研究 [12], [13], [14] では，ニューラルネットワークを用いたハードウェアトロイ識別手法に焦点を当て，組合せ回路と順序回路の両方において，AE に相当する改変を適用したハードウェアトロイ回路の識別率が低下することを確認している．機械学習によるハードウェアトロイ識別を実用化するにあたって，AE 攻撃にも堅牢な識別器を生成する技術を確立する必要がある．

本稿では，機械学習を用いたハードウェアトロイ識別に対する AE 攻撃に関して，データ拡張と敵対的学習の 2 つの手法を採用し，攻撃に堅牢な識別器を生成できるかを実証及び検討する．本稿は，以下のように構成される．2 章で，機械学習を用いたハードウェアトロイ識別および一般的なデータ拡張と敵対的学習の技術について紹介する．3 章で，ハードウェアトロイ識別におけるデータ拡張と敵対的学習の手法を提案する．4 章で，提案手法を適用した識別器によるハードウェアトロイ識別の評価実験の結果を示す．最後に 5 章で，本稿をまとめる．

2. 既存手法

本章では，機械学習を用いたハードウェアトロイ識別とそれに対する AE 攻撃に関してまとめる．また，機械学習分野における学習データを増やす手法であるデータ拡張 (data augmentation) と AE に対しても堅牢な識別器を生成する手法である敵対的学習 (adversarial training) を紹介する．

2.1 節で，機械学習を用いたハードウェアトロイ識別の既存手法とこれに対する AE 攻撃の手法を紹介する．2.2 節で，データ拡張について紹介する．2.3 節で，敵対的学習について紹介する．

2.1 機械学習を用いたハードウェアトロイ識別の既存手法と AE 攻撃

本節では，機械学習を用いたハードウェアトロイ識別の既存手法とこの手法に対する AE 攻撃について紹介する．

2.1.1 機械学習によるハードウェアトロイ識別の既存手法

IC 製品からハードウェアトロイの脅威を取り除くには，初期の段階で検出することが効果的である．製造工程の初期工程である設計段階でハードウェアトロイを識別するには，設計情報から何らかの特徴を抽出する方法が挙げられる．論理レベルのネットリストを対象に，ハードウェアトロイ回路の回路構成上の特徴からネットをスコア付けし，ハードウェアトロイを識別する手法 [6] がある．これを発展させ，近年ではランダムフォレストやニューラルネットワークなどの機械学習を用いたハードウェアトロイ識別手法の研究 [7], [15], [16], [17] も提案されている．文献 [15] では，各ネットから前後のフリップフロップ，マルチプレクサやプライマリ入出力までの距離を特徴量として抽出する．識別の結果は，TPR は平均で 84.8%，最大で 100%である．

2.1.2 ハードウェアトロイ識別に対する AE 攻撃

2.1.1 項で示したハードウェアトロイ識別手法の中で用いられるニューラルネットワークは，一般に AE と呼ばれる識別器に誤識別を引き起こさせる攻撃手法 (AE 攻撃) が存在している [8]．AE を生成する際には，ニューラルネットワークの損失関数の値が大きくなるように，摂動と呼ばれるノイズを加える．この摂動により，対象のデータは正しい分類のサンプル集団からの距離が遠くなり，別のクラスへと分類されるようになる．

ニューラルネットワークを用いたハードウェアトロイ識別に対しても，AE 攻撃が成立することが先行研究 [18] で確認されている．回路においても画像においても，摂動を加えて AE を生成するという点では共通している．計算された AE を入力に反映することは，画像では容易である．一方で，回路においては，任意の改変を適用することは困難で，計算結果を満たすような回路を生成できないことが多い．したがって，回路における改変では，あらかじめ改変パターンを作成しておき，いずれかを選択して適用するという手法を取る [12]．この改変の前後では，回路の論理的等価性は保たれる．改変パターンの適用候補は多数あるため，改変量評価値 (Modification Evaluating Value, MEV) という評価指標を用いて，改変による回路への影響を抑えた最適な改変が選択される．実際の AE 攻撃では，MEV にもとづいて改変を評価し，最適な改変を選択して適用することを複数回繰り返すことが想定されている．

2.2 データ拡張

機械学習において精度の高い識別器を生成するには，十分な学習データが必要となる．そこで，学習データを増やすのに用いられる手法の 1 つがデータ拡張 (data augmentation) である．データ拡張は，ベースとなる元の学習データを加工してデータセットに追加し，データ量を拡張させる手法である．学習器の過学習防止に有効であり，ニューラルネットワークを用いた機械学習の様々な分野で広く使われる．例えば，画像処理分野では，元画像に移動，切り抜き，拡大縮小，回転，反転，ねじれ，色空間の変換などを加えてデータセットを拡張する [19], [20]．自然言語処理分野では，元の文章に単語の置換，挿入，入れ替え，削除などの変更を加えてデータセットを拡張する [21]．データ拡張を採用することで，識別器の全体的精度向上が期待できる．

2.3 敵対的学習

敵対的学習 (adversarial training) は，AE を予め学習データに加えておくことで，識別器の AE への耐性向上を図るものである．攻撃への耐性向上だけでなく，通常の入力に対しても識別制度の向上が見込める手法である．学習の各ステップで新たに AE を生成して学習を進める [8], [22]．AE 及び敵対的学習の研究は画像識別分野で研究が盛んであり，他分野への応用も検討されている [23], [24], [25]．

敵対的学習において，画像の場合，学習ステップごとに AE を算出して生成し，学習データとして与える．一方，回路の場合，毎回 AE を生成して学習させる点は共通するが，AE 生成時には改変パターンを適用したネットリストの候補群から，最適なサンプルを選択するという手法を取る．

3. 提案手法

本章では，ハードウェアトロイ識別器に AE の耐性を持たせる手法として，ハードウェアトロイ識別におけるデータ拡張と敵対的学習の手法を提案する．

3.1 節で，ハードウェアトロイ識別におけるデータ拡張を提案する．3.2 節で，ハードウェアトロイ識別における敵対的学習を提案する．

3.1 ハードウェアトロイ識別におけるデータ拡張

一般的なデータ拡張は，2.2 節で紹介した通り，画像を例にすると，拡大縮小，回転及び反転などの処理を加えたデータを学習データセットに加えるものである．回路において，上記の画像における操作に直接対応するものは存在しないので，回路用に独自に定義する．

本節では，回路用のデータ拡張用の改変として，回路をゲートの挿入などで論理的に等価な別の回路に改変することを提案する．ここでの改変は，先行研究 [18] における AE 生成と同等である．具体的な改変は 4.1.1 項で示す．

データ拡張を用いたハードウェアトロイ識別のアルゴリズムを Algorithm 1 に示す．このアルゴリズムでは，ベースとなる回路に適用可能な改変パターンを全通り適用し，

Algorithm 1 データ拡張のアルゴリズム．

Inputs: The circuit infected with a hardware Trojan G, AE patterns P, the maximum number of iterations K
Output: A hardware Trojan classifier f

```
D_train ⇐ G
G_base ⇐ G
for i = 1 to K do
  for j = 1 to i do
    G_AE ⇐ ∅
    for all g ∈ G_base do
      for all p ∈ P that can be applied to g do
        for all Applicable gate in hardware-Trojan part of g do
          Apply p to the gate and generate the modified circuit g'
          G_AE ⇐ G_AE ⋃ {g'}
        end for
      end for
    end for
    G_base ⇐ G_AE
    D_train ⇐ D_train ⋃ G_AE
  end for
end for
Make classifier f with training dataset D_train
return f
```

その全てを学習用のデータセットに加える．1 つの改変パターンでも適用可能な箇所は複数存在しうるため，改変適用後の回路は複数生成されうる．1 つの回路あたりに加える改変の最大回数を K とする．学習に用いるハードウェアトロイを含む回路 $g \in G$ に関して，改変が適用可能な箇所の全数を m_g とする．このアルゴリズムで生成されるネットリストの数は，次の式 (1) で表す通りである．

$$\sum_{g \in G} \sum_{i=1}^{K} \binom{m_g}{i} \tag{1}$$

3.2 ハードウェアトロイ識別における敵対的学習

敵対的学習では，学習を進める都度，AE を生成し学習する．本節では，ハードウェアトロイ識別における敵対的学習のアルゴリズムを提案する．

Algorithm 2 に提案するアルゴリズムを示す．アルゴリズム内では，過去に提案した改変パターンの評価指標である MEV と MEV を利用した AE 生成手法を用いる [18]．ベースとなる回路から一定数の R をランダムに選出し，上記手法で K [回] 改変を適用した最も効果的な AE を生成する．その後，生成された AE をもともとの学習用データセットと差し替える．このデータセットを用いて，新しく学習し新たな学習器を生成する．この一連の処理を E [回] 繰り返す．前回までの学習で弱い部分を補強するように，データセットを差し替えて学習を進めるようになっている．

Algorithm 2 敵対的学習のアルゴリズム．

Inputs: Learned classifier f_{init}, the circuit infected with a hardware Trojan G, AE patterns P, the number of modification K, the number of epochs E, the number of replacement R
Output: A hardware Trojan classifier f

```
f ⇐ f_init
for i = 1 to E do
  D_train ⇐ G
  G_AE ⇐ ∅
  Select R circuits G_sub randomly from G
  for all g ∈ G_sub do
    Generate an adversarial example g' for f from g with K patterns P' ⊂ P evaluated by MEV
    G_AE ⇐ G_AE ⋃ {g'}
  end for
  for all g ∈ G_AE do
    Replace the circuit d ∈ D_train with g where base circuit of g is same to d
    Make new classifier f with training dataset D_train
  end for
end for
return f
```

4. 評価実験

本章では，3 章で提案した手法をもとに，ベンチマークを用いて評価実験した結果を示す．4.1 節で，実験結果を示す．4.2 節で，実験結果をもとに考察する．

4.1 実験結果

本節では，評価実験の実験条件を整理したのち，実験結果を示す．

4.1.1 実験条件

本稿の実験では，Trust-HUB[26], [27], [28] で公開されているベンチマークのうち，15 種類の論理レベルのネットリストを利用した．**表 2** に利用したハードウェアトロイのベンチマークを示す．15 種類のベンチマークのうち，RS232-T1200 を識別用として利用し，残りの 14 種類を学習用として利用した．RS232-T1200 のハードウェアトロイ部分の回路図を**図 1** に示す．

実験で使用するニューラルネットワークは，入力層 51 ユニット，中間層 3 層 (各 200, 100, 50 ユニット)，出力層 2 ユニットで構成され，活性化関数にはシグモイド関数を用いた．実験には，CPU として Intel Xeon E7-8855 v4 (@2.10GHz × 112)，メモリ 1TB を搭載したサーバを使用した．特徴量は，文献 [7] で提案されている 51 種類の特徴量をすべて用いる．

ハードウェアトロイに適用する改変パターンは，これまでに提案しているものや新たに追加したものをあわせて，$t1$–$t13$, dff, dffinv, mux2 の 16 種類である [14], [18]．16 種類の改変パターンとその適用先を**表 1** に示す．適用可能な改変パターンは，適用先の回路によって異なる．図 1 の回

表 1: 改変パターンと適用可能箇所.
Table 1 Modification patterns and their applicable places.

パターン	適用可能箇所
$t1$	4 入力 OR
$t2$	4 入力 OR
$t3$	2 入力 OR
$t4$	WIRE
$t5$	2 入力 AND
$t6$	2 入力 NAND
$t7$	4 入力 AND
$t8$	2 入力 NOR
$t9$	4 入力 NOR
$t10$	3 入力 AND
$t11$	3 入力 NAND
$t12$	3 入力 NOR
$t13$	2 入力 NAND
dff	D フリップフロップ
dffinv	D フリップフロップ
mux2	2 入力マルチプレクサ

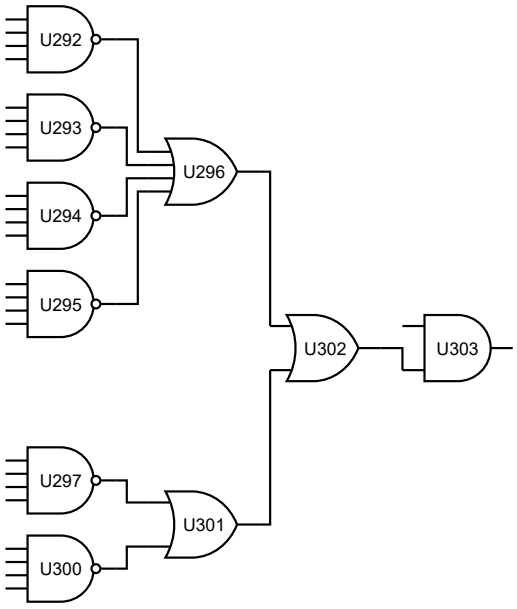

図 1: RS232-T1200 のハードウェアトロイ回路.
Fig. 1 Hardware Trojan circuit of RS232-T1200.

路のゲート U296 に改変パターン $t1$ を適用した回路を**図 2** に示す.

本稿の実験では，初期の評価のため，同一の回路に改変を加える回数 K は $K = 1$ 回とする．そのため，改変が複数回適用された回路は学習用のデータセットに含まれない．また，AE 耐性の評価時に識別させる AE の改変回数も 1 回とする．データ拡張では，Algorithm 1 に示す通り，学習用のデータセット 14 種に改変パターンを適用し，追加で 239 のネットリストを得た．敵対的学習では，Algorithm 2 に示す通り，学習を繰り返す回数 E を $E = 5$ 回，データセットの中で差し替える数 R をネットリスト数の半分の $R = 7$ として実験した.

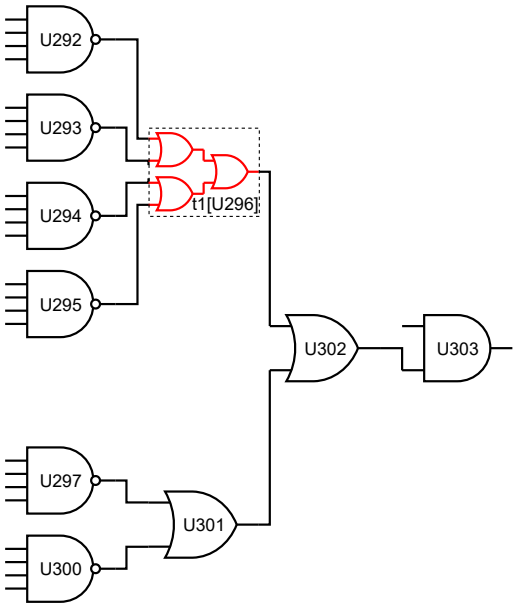

図 2: RS232-T1200 のハードウェアトロイ部分に改変パターン $t1$ を適用した回路 (赤色部).
Fig. 2 Hardware Trojan circuit of RS232-T1200 with modification pattern $t1$ (red part).

表 2: 実験で使用したベンチマークの一覧.
Table 2 List of benchmarks utilized in the experiment.

ベンチマーク	ノーマルネット数	トロイネット数
RS232-T1000	283	36
RS232-T1100	284	36
RS232-T1200	289	34
RS232-T1300	287	29
RS232-T1400	273	45
RS232-T1500	283	39
RS232-T1600	292	29
s15850-T100	2419	27
s35932-T100	6407	15
s35932-T200	6405	12
s35932-T300	6405	37
s38417-T100	5798	12
s38417-T200	5798	15
s38417-T300	5841	44
s38584-T100	7353	19

4.1.2 結果

実験では，RS232-T1200 に AE を適用した回路を識別する際に，データ拡張と敵対的学習を用いた．**表 3** に，元の識別器，敵対的学習を 5 回適用した識別器，データ拡張を適用した識別器それぞれでの TPR (True Positive Rate) と TNR (True Negative Rate) を示す．ここで，TPR はトロイネットのうち正しくトロイネットと識別された割合で，TNR はノーマルネットのうち正しくノーマルネットと識別された割合である．表 3 に示す通り，TPR の最小値は対策前の識別器では 80.56%だったが，敵対的学習では 94.44%，データ拡張では 97.22%となり改善した．一

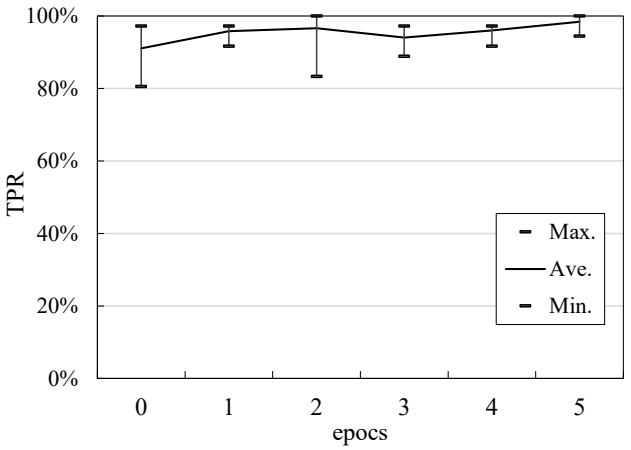

図 3: 5 回の敵対的学習時の TPR の変化.

Fig. 3 Transition of TPR over five times adversarial training.

方，TNR の最小値は対策前の 97.58%から，敵対的学習では 93.08%，データ拡張では 95.50%へと低下した．

次に，敵対的学習を 5 回繰り返す中での TPR の最小値，平均値，最大値の変化を**図 3** に示す．ここでの値は，AE を適用していない元の回路 (表 3 中の original) を除いて算出している．TPR の平均値は，対策前から 2 回学習時まで向上し，3 回学習時に一度低下しているが，4 回学習時と 5 回学習時で再度改善している．最終的に得られた 5 回学習時の識別器は，TPR の平均値は一番高く，最大値と最小値の幅も小さくなっている．

4.2 考察

データ拡張と敵対的学習を適用した識別器では，いずれも性能が向上しているものの TPR や TNR の傾向が異なる．TPR の平均値は，敵対的学習 5 回時の結果が 98.41%であり，データ拡張の 97.22%より優れている．また，最大値も敵対的学習時は 100.00%に達するものが 14 種のネットリスト中 9 種あり，100.00%に達するものがないデータ拡張より優れているといえる．一方で，最小値は，敵対的学習は $t4$ に弱く 94.44%を出しており，データ拡張の 97.22%を下回る．最大や平均が良い場合でも，弱点を突かれてしまえば突破されかねないため，敵対的学習はこの点でデータ拡張より劣る．

また，TNR の観点では，敵対的学習は平均で 93.56%，データ拡張は 95.71%となっている．敵対的学習の方が，ノーマルのネットをトロイネットと判定する傾向が強いといえる．

次に，学習時のコストの観点で比較する．敵対的学習は，14 種のベンチマークの一部に改変を適用して入れ替えながら学習することを複数回繰り返す．一度に扱うネットリストの数は少ないが，学習を重ねる必要がある．毎回の改変を生成する工程は，MEV を基準として最適なものを効率的に生成している．1 つの回路当たりの改変の回数を増やした場合，AE 生成時の手法内で改変を選択する回数が増えることによる多少の処理の増加がある．データ拡張は，適用できる改変を全て列挙し，学習データとして利用する．全ての改変されたネットリストを学習するため，学習にかかる時間が増大する．1 つの回路当たりの改変の回数を増やした場合，全通りの改変を生成する時間の増加に加えて，学習用データセットの増加に伴う学習時間の増加がある．

本稿の実験では，敵対的学習時には学習回数を 5 回と設定した．実験結果から判断するに，識別性能は十分に向上しており，学習が一定程度収束しているはみなせるが，さらに学習を続けた場合の結果を今後確認する必要があるといえる．敵対的学習とデータ拡張は一長一短がある．多くのトロイネットを正しく識別するという点と，AE を選択的に効率良く生成でき，一度の学習時間が短い点では敵対的学習が優れている．最悪の攻撃の場合でも正しく識別するという点ではデータ拡張が優れている．ただし，最初の改変パターンの作成で，全通りの改変を生成しなければならないため，1 つの回路に適用する改変の数が増えた時にコスト増加が顕著である．どちらを採用するかは，学習にかけられる時間や扱えるネットリストの量に応じて判断する必要がある．今回は 1 回の改変しか適用していないため，改変を複数回適用した場合の 2 つの手法の性能変化などを今後調査する必要がある．

5. おわりに

本稿では，機械学習を用いたハードウェアトロイ識別に対する AE 攻撃に関して，データ拡張と敵対的学習の 2 つの手法を提案した．また，提案手法に基づき，攻撃に堅牢な識別器を生成できるかを実証実験をした．実験の結果，いずれの手法でも AE 攻撃に対する識別器の耐性を高めることに成功した．本稿の実験では，識別用に利用したベンチマークや改変を適用した回数が限られている．今後，他のベンチマークや改変や学習の回数を変えて対策手法の評価及び検討を継続していく．

参考文献

[1] Rostami, M., Koushanfar, F., Rajendran, J. and Karri, R.: Hardware security: threat models and metrics, *Proc. International Conference on Computer-Aided Design (ICCAD)*, pp. 819–823 (2013).

[2] Francq, J. and Frick, F.: Introduction to hardware Trojan detection methods, *Proc. 2015 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, EDAA, pp. 770–775 (2015).

[3] Xiao, K., Forte, D., Jin, Y., Karri, R., Bhunia, S. and Tehranipoor, M.: Hardware trojans: lessons learned after one decade of research, *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, Vol. 22, No. 1, pp. 1–23 (2016).

[4] Liu, B. and Qu, G.: VLSI supply chain security risks and mitigation techniques : A survey, *Integration, the VLSI Journal*, Vol. 55, pp. 438–448 (2016).

表 3: データ拡張と敵対的学習による TPR と TNR の変化 (RS232-T1200).
Table 3 Changes in TPR and TNR with data augmentation and adversarial training (RS232-T1200).

回路名 RS232-T1200 パターン [改変箇所]	元の識別器		敵対的学習 5 回後の識別器		データ拡張 適用後の識別器	
	TPR	TNR	TPR	TNR	TPR	TNR
original	97.06%	97.58%	100.00%	93.08%	97.06%	95.50%
t1[U296]	94.44%	97.58%	100.00%	94.12%	97.22%	95.85%
t2[U296]	86.11%	97.58%	97.22%	94.12%	97.22%	95.85%
t3[U301]	97.22%	97.59%	100.00%	93.13%	97.22%	95.53%
t3[U302]	94.44%	97.59%	100.00%	93.81%	97.22%	95.88%
t4[U296]	80.56%	97.58%	94.44%	94.12%	97.22%	95.85%
t4[U301]	94.44%	97.58%	97.22%	93.08%	97.22%	95.50%
t4[U302]	80.56%	97.58%	94.44%	94.12%	97.22%	95.85%
t5[U303]	94.44%	97.59%	100.00%	93.81%	97.22%	95.88%
t6[U292]	86.11%	97.58%	94.44%	93.08%	97.22%	95.50%
t6[U293]	88.89%	97.58%	100.00%	93.77%	97.22%	95.85%
t6[U294]	91.67%	97.58%	100.00%	93.43%	97.22%	95.85%
t6[U295]	97.22%	97.58%	100.00%	93.08%	97.22%	95.50%
t6[U297]	91.67%	97.58%	100.00%	93.08%	97.22%	95.50%
t6[U300]	97.22%	97.58%	100.00%	93.08%	97.22%	95.50%

[5] Chakraborty, R. S., Narasimhan, S. and Bhunia, S.: Hardware Trojan: threats and emerging solutions, *Proc. International High-Level Design Validation and Test Workshop (HLDVT)*, pp. 166–171 (2009).

[6] Oya, M., Shi, Y., Yanagisawa, M. and Togawa, N.: A score-based classification method for identifying hardware-Trojans at gate-level netlists, *Proc. 2015 Design, Automation & Test in Europe Conference & Exhibition*, pp. 465–470 (2015).

[7] Hasegawa, K., Yanagisawa, M. and Togawa, N.: Trojan-feature extraction at gate-level netlists and its application to hardware-Trojan detection using random forest classifier, *Proc. IEEE International Symposium on Circuits and Systems*, (online), DOI: 10.1109/ISCAS.2017.8050827 (2017).

[8] Goodfellow, I. J., Shlens, J. and Szegedy, C.: Explaining and harnessing adversarial examples, *Proc. 2015 International Conference on Learning Representations (ICLR)* (2015).

[9] Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. and Fergus, R.: Intriguing properties of neural networks, *arXiv preprint arXiv:1312.6199*, pp. 1–10 (online), DOI: 10.1021/ct2009208 (2013).

[10] Akhtar, N. and Mian, A.: Threat of adversarial attacks on deep learning in computer vision: a survey, *IEEE Access*, pp. 14410–14430 (2018).

[11] Eykholt, K., Evtimov, I., Fernandes, E., Li, B., Rahmati, A., Xiao, C., Prakash, A., Kohno, T. and Song, D.: Robust Physical-World Attacks on Deep Learning Models, *Computing Research Repository (CoRR)*, Vol. abs/1707.0 (online), DOI: 10.1109/CVPR.2018.00175 (2017).

[12] Nozawa, K., Hasegawa, K., Hidano, S., Kiyomoto, S., Hashimoto, K. and Togawa, N.: Adversarial Examples for Hardware-Trojan Detection at Gate-Level Netlists, *Proc. 2019 International Workshop on Attacks and Defenses for Internet-of-Things (ADIoT)*, pp. 1–18 (2019).

[13] 野澤康平，長谷川健人，披田野清良，清本晋作，橋本和夫，戸川望：ニューラルネットワークを用いたハードウェアトロイ識別に対する敵対的サンプル攻撃の実証評価，信学技報，HWS2019-64, ICD2019-25, Vol. 119, No. 260, 大阪，pp. 41–46 (2019).

[14] Nozawa, K., Hasegawa, K., Hidano, S., Kiyomoto, S., Hashimoto, K. and Togawa, N.: Application of Adversarial Examples for Hardware Trojan Detection to Sequential Circuits, *Proc. 2020 Symposium on Cryptography and Information Security (SCIS)* (2020).

[15] Hasegawa, K., Yanagisawa, M. and Togawa, N.: Hardware Trojans classification for gate-level netlists using multi-layer neural networks, *Proc. 2017 IEEE 23rd International Symposium on On-Line Testing and Robust System Design (IOLTS)*, pp. 227–232 (2017).

[16] Inoue, T., Hasegawa, K., Yanagisawa, M. and Togawa, N.: Designing hardware Trojans and their detection based on a SVM-based approach, *Proc. International Conference on ASIC*, pp. 811–814 (2018).

[17] Dong, C., He, G., Liu, X., Yang, Y. and Guo, W.: A multi-layer hardware trojan protection framework for IoT chips, *IEEE Access*, Vol. 7, pp. 23628–23639 (2019).

[18] Nozawa, K., Hasegawa, K., Hidano, S., Kiyomoto, S., Hashimoto, K. and Togawa, N.: Adversarial Examples for Hardware-Trojan Detection at Gate-Level Netlists, *Computer Security*, Cham, Springer International Publishing, pp. 341–359 (2020).

[19] Perez, L. and Wang, J.: The Effectiveness of Data Augmentation in Image Classification using Deep Learning, pp. 1–6 (2017).

[20] Shorten, C. and Khoshgoftaar, T. M.: A survey on Image Data Augmentation for Deep Learning, *Journal of Big Data*, Vol. 6, No. 1, p. 60 (online), DOI: 10.1186/s40537-019-0197-0 (2019).

[21] Wei, J. and Zou, K.: EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks, *Proc. the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th*

International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Stroudsburg, PA, USA, Association for Computational Linguistics, pp. 6381–6387 (online), DOI: 10.18653/v1/D19-1670 (2019).

[22] Kurakin, A., Goodfellow, I. J. and Bengio, S.: Adversarial Machine Learning at Scale, *Proc. 2017 5th International Conference on Learning Representations (ICLR)*, OpenReview.net (2017).

[23] Kurakin, A., Goodfellow, I. J. and Bengio, S.: Adversarial examples in the physical world, *Proc. 2017 International Conference on Learning Representations (ICLR)* (2017).

[24] Dai, H., Li, H., Tian, T., Huang, X., Wang, L., Zhu, J. and Song, L.: Adversarial attack on graph structured data, *Proc. International Conference on Machine Learning (ICML)* (2018).

[25] Jia, R. and Liang, P.: Adversarial examples for evaluating reading comprehension systems, *Proc. of the 2017 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, pp. 2021–2031 (2017).

[26] : Trust-HUB.

[27] Shakya, B., He, T., Salmani, H., Forte, D., Bhunia, S. and Tehranipoor, M.: Benchmarking of hardware trojans and maliciously affected circuits, *Journal of Hardware and Systems Security*, Vol. 1, No. 1, pp. 85–102 (2017).

[28] Salmani, H., Tehranipoor, M. and Karri, R.: On design vulnerability analysis and trust benchmarks development, *2013 IEEE 31st International Conference on Computer Design (ICCD)*, pp. 471–474 (2013).