Vol.2019-ARC-235 No.51
Vol.2019-SLDM-187 No.51
Vol.2019-EMB-50 No.51
2019/3/18

周辺ネットの特徴量を考慮した二段階のニューラルネットワークによるハードウェアトロイ検出手法

井上 智貴[1,a)]　長谷川 健人[1]　戸川 望[1]

概要：近年，急速な IoT (Internet of Things) 化により，日常の様々なものに組込みハードウェアが利用されている．組込みハードウェアの需要増加によりハードウェアベンダは設計や製造の一部を第三者に委託するようになっている．これにより悪意の回路 (ハードウェアトロイ) がハードウェアに挿入される危険性が増大している．本稿では，ゲートレベルネットリスト中の各ネットがハードウェアトロイを構成するネットか否かを識別することを目的に，ハードウェアトロイの局所性をもとに周辺ネットの特徴量を考慮した，Neural Network (NN) によるハードウェアトロイ検出手法を提案する．提案手法は二段階の学習と識別により構成される．第一段階として，11 個の特徴量を抽出，学習し，識別により各ネットの Trojan_prob (対象のネットがハードウェアトロイの一部である疑いの度合い) を取得する．Trojan_prob をもとに周辺のネットの Trojan_prob を考慮した新たな特徴量を抽出する．第二段階として，新たに抽出した 4 つの特徴量を加えた 15 の特徴量を学習し，識別により最終的にネットをトロイネットか否かに分類する．実験の結果，平均 TPR 83.6%，平均 TNR 96.5%を示し，第一段階だけの識別に比べ平均 TPR は 4.31 ポイント向上した．

キーワード：ハードウェアトロイ，デザインタイム，ゲートレベルネットリスト，機械学習，ニューラルネットワーク

1. はじめに

近年，急速な IoT (Internet of Things) 化に伴い，身の回りの様々なものに組込みハードウェアが利用されるようになった．組込みハードウェアの需要増加に伴い，生産をより安価に抑えるため製造拠点は国際化し，設計や製造の一部は第三者に委託されるようになっている．第三者に委託された設計や製造において，ハードウェア製品にハードウェアトロイ (ハードウェアに挿入された悪意の回路) が挿入される危険が指摘されている．ハードウェアトロイの機能は機密情報の漏洩や論理値の改竄，システムの破壊や電力の異常消費など多岐にわたる [3]．設計や製造の外部委託の増加により，攻撃者にとってハードウェアトロイの挿入が容易になっている．さらに，ハードウェアの性能向上による機能の多様性から，ハードウェアトロイは個人や企業，国にとって大きな脅威となっている．ハードウェアの中身はブラックボックスであることが多く，ハードウェアの信頼性はハードウェアベンダへの信頼性に依存している現状を受け，ハードウェアトロイの脅威とその対策に対する研究が盛んとなっている [3]．本稿では，ハードウェア設計情報のうち，ゲートレベルネットリストに挿入されたハードウェアトロイの検出に着目する．

ハードウェア設計情報に挿入されるハードウェアトロイの一般的な構造を図 1 に示す．ハードウェアトロイはトリガ回路とペイロード回路により構成される．トリガ回路は，回路の内部状態がトリガ条件を満足しているかどうかを判定する．回路の内部状態がトリガ条件を満足している場合，トリガ回路はペイロード回路をアクティブにする．ペイロード回路は実際に機密情報の漏洩やシステムの破壊などの悪意の機能を発現する．

ハードウェアトロイ検出手法は近年数多く提案されている．論理テストによる検出手法 [1,2] では，様々な入力信号に対し出力信号を観測，解析し，ハードウェアトロイの有無を識別する．論理テストを用いた手法は膨大な入力信号を試す必要があり，時間がかかるという点が課題となっている．サイドチャネル解析による検出手法 [7–9] では，ノイズや消費電力など，外部から得られる物理的情報 (サ

[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
[a)] tomotaka.inoue@togawa.cs.waseda.ac.jp

Vol.2019-ARC-235 No.51
Vol.2019-SLDM-187 No.51
Vol.2019-EMB-50 No.51
2019/3/18

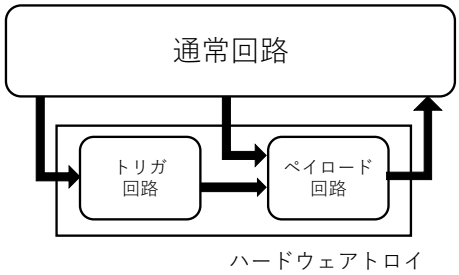

図 1: ハードウェアトロイの一般的な構造.

イドチャネル情報) を元にハードウェアトロイを検出する．ノイズや消費電力などの物理的情報は製造された IC チップから観測する．機械学習を用いた検出手法 [4, 5, 11] では，ハードウェアトロイの挿入位置が既知である回路を学習し，設計段階における IC チップの設計情報をもとにハードウェアトロイを検出する．

ハードウェアトロイは日々進化しており，既知のハードウェアトロイに加え未知のハードウェアトロイも検出できることが望ましい．文献 [5] では，Neural Network (NN) を用いたハードウェアトロイ検出手法を提案している．ゲートレベルのネットリストに対し，各ネットがハードウェアトロイを構成するネット (トロイネット) か否か (ノーマルネット) を識別している．大部分のトロイネット検知に成功しているものの，周辺ネットがトロイネットであるにも関わらずノーマルネットに分類されているネットが存在する．ここで，ネットの周辺情報を利用することで，周辺ネットがトロイネットであるにも関わらず，ノーマルネットに分類されているネットを正しく分類し直すことが期待できる．

本稿では，ハードウェアトロイの局所性をもとに周辺ネットの特徴量を考慮したハードウェアトロイ検出手法を提案する．ハードウェアトロイは，小規模に構成されたトロイ回路がノーマル回路に挿入されるため，局所的に集中して存在する．提案手法ではこの局所性を利用し，二段階の識別によりトロイネットとノーマルネットを識別する．

本稿の貢献を以下に示す．

(1) Trojan_prob なる対象ネットがハードウェアトロイの一部である疑いの度合いをもとに 4 種類の特徴量を提案する．
(2) (1) を利用して，ハードウェアトロイの局所性を利用したハードウェアトロイ検出手法を提案する．
(3) Trust-HUB のベンチマークに対して適用し，交差検証したところ，平均 TPR 83.6%，平均 TNR 96.5%を達成した．

本稿の構成を以下に示す．2 章ではゲートレベルネットリストにおけるハードウェアトロイ検出手法の現状を説明する．3 章ではハードウェアトロイの局所性に着目し，これをよく表す 4 つの特徴量を提案する．4 章ではハードウェアトロイの局所性をもとに周辺ネットの特徴量を考慮した，NN を用いたハードウェアトロイ検出手法を提案する．5 章では提案手法によるハードウェアトロイ識別実験の結果を示す．6 章で本稿をまとめる．

2. ゲートレベルネットリストにおけるハードウェアトロイの検出

IC チップ設計段階におけるハードウェアトロイ検出ができれば，製造後におけるハードウェアトロイ検出に比べコストや時間がかからない．例えば，設計段階では 1 つのゲートレベルネットリストが検出対象であるのに対し，製造後では全製品が検出対象となり，検出対象の数が大きく異なる．また，IC チップやゲートレベルネットリストといったリソースにハードウェアトロイが挿入されていないことを証明するのは非常に困難であるため，ゴールデンリソース (ハードウェアトロイが挿入されていないことが保証されたリソース) を使用せず，未知のハードウェアトロイの検出が期待できる手法であることが強く求められる [10]．そこで本稿では，IC チップの設計段階に注目し，未知のハードウェアトロイ検出に対する有効な手段として，機械学習を用いたハードウェアトロイ検出手法に注目する．

文献 [4] で，SVM (Support Vector Machine) を用いたハードウェアトロイ検出手法が提案されている．ゲートレベルネットリストをもとに，各ネットのプライマリ入力までの距離などを含む 5 つの特徴量を抽出し，SVM を用いてネットをノーマルネットとトロイネットに分類する．文献 [5] では，文献 [4] の 5 個の特徴量にもとづく NN を用いたハードウェアトロイ検出手法が提案されている．NN を用いたハードウェアトロイ検出手法は SVM を用いたハードウェアトロイ検出手法と比較して，実験の結果から得られた平均 True Positeve Rate (TPR) においてほぼ同等の結果を示し，平均 True Negative Rate (TNR) においてより高い結果を示している．セキュリティの観点では TPR の向上が重要である一方で，実用性を考慮すると TNR も向上させる必要がある．この点において，NN を用いたハードウェアトロイ検出手法 [5] は SVM を用いたハードウェアトロイ検出手法 [4] よりも優れている．さらに文献 [6] では，文献 [5] で利用されたハードウェアトロイ検出のためにネットから抽出する特徴量を改良し，より高い TPR と TNR を達成している．しかし，高い TPR を達成し，多くのトロイネットの検出に成功しているが，周辺ネットがトロイネットであるにも関わらず，ノーマルネットに分類されているネットが存在するという課題がある．また，逆に周辺ネットがノーマルネットであるにも関わらず，トロイネットに分類されているネットも存在している．周辺のネットの情報を考慮することができれば，ネットを正しく分類し，より高い精度で識別することが期待できる．

本稿では NN を用いたハードウェアトロイ検出手法 [6] を発展させ，ハードウェアトロイの局所性を利用した NN

Vol.2019-ARC-235 No.51
Vol.2019-SLDM-187 No.51
Vol.2019-EMB-50 No.51
2019/3/18

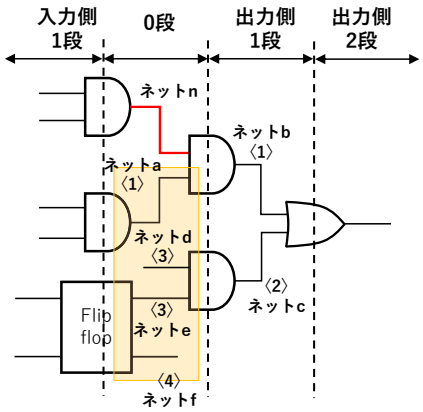

図 2: Trojan_0 が参照するネット．

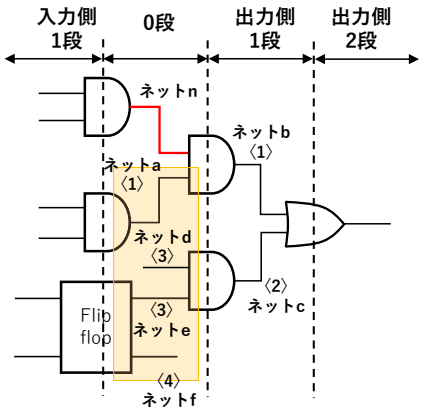

図 3: Trojan_in_1 が参照するネット．

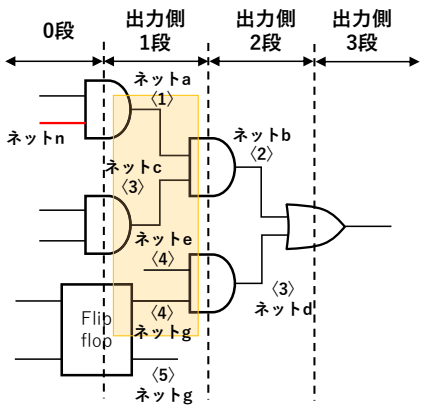

図 4: Trojan_out_1 が参照するネット．

によるハードウェアトロイ検出手法を提案する．提案手法は二段階の学習と識別により構成される．第一段階として，11 個の特徴量を抽出，学習し，識別により各ネットの Trojan_prob (対象のネットがハードウェアトロイの一部である疑いの度合い) を取得する．Trojan_prob をもとに周辺のネットの Trojan_prob を考慮した新たな特徴量を抽出する．第二段階として，新たに抽出した 4 つの特徴量を加えた 15 の特徴量を学習し，識別により最終的にネットをトロイネットか否かに分類する．

3. ハードウェアトロイの局所性と特徴量

本章では，ハードウェアトロイの局所性を考慮したハードウェアトロイ検出手法において利用する特徴量を提案する．

ハードウェアトロイは小規模に構成された回路がノーマルネット中に埋め込まれるため，トロイネットは局所的に存在する．機械学習を用いたハードウェアトロイ識別手法から得られた識別結果において，周辺のネットがトロイネットと識別されたにも関わらず，あるネットがノーマルネットと識別された場合，そのネットはトロイネットである可能性が極めて高い．この特性を利用することで，たとえ一度の機械学習を用いたハードウェアトロイ識別で一部のトロイネットを誤ってノーマルネットと識別していたとしても，少数の真のトロイネットを正しくトロイネットと識別できれば，その情報をもとにさらに周辺のネットの情報を考慮して取り逃したトロイネットを正しくトロイネットと識別できる．

NN を利用したハードウェアトロイ検出手法 [5] は，ネットリストから抽出した 11 個の特徴量により NN を用いて，ネットをトロイネットかノーマルネットに分類する．文献 [5] で利用されるニューラルネットワークは，2 個のユニットからなる出力層をもつ．それぞれのユニットはノーマルネットとトロイネットに対応しており，学習時にはノーマルネットを (1, 0)，トロイネットを (0, 1) として学習する．識別時には，2 つのユニットの出力値を比較し，大きい方を識別結果とみなすことができる．本稿では，ノーマルネットに対応するユニットの出力値を Normal_prob，トロイネットに対応するユニットの出力値を Trojan_prob と呼ぶ．識別結果がノーマルネットのときは Normal_prob の値が，トロイネットのときは Trojan_prob の値が大きくなるため，それぞれノーマルネットらしさ，トロイネットらしさの度合いを表す値と見なすことができる．

今，[5] により，第一段階の識別を行い，各ネットに対して識別結果，すなわち，Normal_prob と Trojan_prob の値を得たとする．2 章の議論より，機械学習を用いたハードウェアトロイ識別そのままでは周辺のネットの情報を考慮することができない．そこで，Trojan_prob で得られる情報をもとに，周辺のネットがトロイネットか否かを考慮した特徴量を新たに提案する．特に，注目するネットに対し，前後一段以内のネットにトロイネットが存在するか否かが重要であるため，ここではこれらを考慮した以下の 4 つの特徴量を提案する．

(1) Trojan_prob
対象ネットがトロイネットである疑いの度合いを表す． -10 から 10 の値で，値が大きいほど対象ネットがトロイネットである疑いが高いことを示す．

(2) Trojan_0
対象ネットからネット 4 本以内の距離にある同段ネットの Trojan_prob の平均値を Trojan_0 とする．

(3) Trojan_in_1
対象ネットからネット 4 本以内の距離にある入力側で 1 段離れたネットの Trojan_prob の平均値を Trojan_in_1 とする．

(4) Trojan_out_1
対象ネットからネット 4 本以内の距離にある出力側で 1 段離れたネットの Trojan_prob の平均値を Trojan_out_1 とする..

ハードウェアトロイの局所性を特徴量に反映するため，Trojan_0, Trojan_in_1, Trojan_out_1 は対象ネットからネット 4 本以内の距離にあるネットの Trojan_prob を参照する．

Vol.2019-ARC-235 No.51
Vol.2019-SLDM-187 No.51
Vol.2019-EMB-50 No.51
2019/3/18

表 1: 提案手法で用いる 15 個のネット特徴量 [5].

#	特徴量	説明
1	fan_in_4	対象のネットの入力側における 4 段手前のゲート数.
2	fan_in_5	対象のネットの入力側における 5 段手前のゲート数.
3	in_flipflop_4	対象のネットの入力側で 4 段以上離れたフリップフロップ数.
4	out_flipfrop_3	対象のネットの出力側で 3 段以上離れたフリップフロップ数.
5	out_flipflop_4	対象のネットの出力側で 4 段以上離れたフリップフロップ数.
6	in_loop_4	対象のネットの入力側で 4 段以上のループ数.
7	out_loop_5	対象のネットの出力側で 5 段以上のループ数.
8	in_nearest_pin	対象のネットからプライマリ入力までの最小の段数.
9	out_nearest_pout	対象のネットからプライマリ出力までの最小の段数.
10	out_nearest_flipflop	対象のネットの出力側の直近のフリップフロップまでの段数.
11	out_nearest_multiplexer	対象のネットの出力側の直近の MUX までの段数.
12	Trojan_prob	対象ネットがトロイネットである疑いの度合い.
13	Trojan_0	対象ネットからネット 4 本以内の距離にある同段ネットの Trojan_prob の平均値.
14	Trojan_in_1	対象ネットからネット 4 本以内の距離にある入力側で 1 段離れたネットの Trojan_prob の平均値.
15	Trojan_out_1	対象ネットからネット 4 本以内の距離にある出力側で 1 段離れたネットの Trojan_prob の平均値.

例えば，図 2 に Trojan_0 が参照するネットを示す．赤で示されたネット n が対象ネットである．ネット n が入力側で接続している AND ゲートの入力側で接続するネット a，出力側で接続するネット b は，ネット n からネット 1 本分の距離にある．ネット b が入力側で接続している OR ゲートの入力側で接続するネット c は，ネット n からネット 2 本分の距離にある．このようにネット n からの距離を数えると，ネット d，ネット e はネット n からネット 3 本分の距離にあり，ネット f はネット n からネット 4 本分の距離にある．Trojan_0 の設計より，ここで参照するのは黄色で示された範囲にあるネット a,d,e,f となり，これらの Trojan_prob の平均値を取得する．図 3 に Trojan_in_1 が参照するネットを示す．図 2 と同様に，Trojan_in_1 ではネット a,b,d,e,f を参照し，これらの Trojan_prob の平均値を取得する．図 4 に Trojan_out_1 が参照するネットを示す．図 2 と同様に，Trojan_in_1 ではネット a,c,e,g を参照し，これらの Trojan_prob の平均値を取得する．なお，対象ネットからネット 4 本以内の距離に参照するネットが存在しない場合，各特徴量は 0 と定義する．

提案する 4 つの特徴量は第一段階の識別結果をもとに決定されるため，ネットの誤分類を含んでいる．第一段階の識別で多くのトロイネットとノーマルネットを正しく分類でき，トロイネットが誤ってノーマルネットに分類されたとしても，周辺のネットは正しくトロイネットに分類されることが多い．逆に第一段階の識別でノーマルネットが誤ってトロイネットに分類されたとしても，周辺のネットは正しくノーマルネットに分類されることが多い．上記の理由から，誤分類されたネットの周辺のネットは正しく分類されていることが多いので，局所性を反映した特徴量を利用することで，TPR，TNR を向上する．

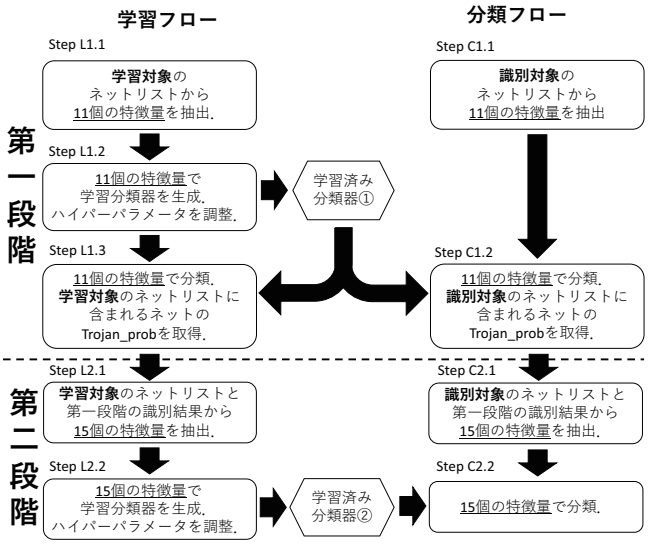

図 5: 局所性を利用したハードウェアトロイ検出手法の流れ.

4. ニューラルネットワークを用いたハードウェアトロイ検出手法

本稿で提案するハードウェアトロイの局所性を利用した NN によるハードウェアトロイ検出手法は対象ベンチマークを二段階で識別する．

表 1 に，提案手法で用いる計 15 個のネット特徴量を示す．表 1 の特徴量 12–15 は 3 章で定義した特徴量である．

図 5 に提案手法の流れを示す．提案手法は，学習フローと識別フローから構成される．学習フローでは予めトロイネットが既知であるネットリストを学習し，分類器を生成する．識別フローでは，学習フローで生成した分類器を用いてネットリスト中の各ネットをノーマルネットかトロイネットに識別する．

Vol.2019-ARC-235 No.51
Vol.2019-SLDM-187 No.51
Vol.2019-EMB-50 No.51
2019/3/18

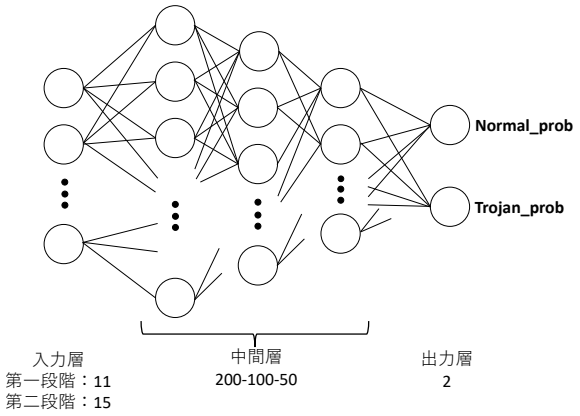

図 6: NN の構造.

表 2: 学習と識別に用いたベンチマーク回路 [13].

ベンチマーク	ノーマルネット数	トロイネット数
RS232-T1000	309	10
RS232-T1100	304	11
RS232-T1200	310	13
RS232-T1300	309	7
RS232-T1400	306	12
RS232-T1500	311	11
RS232-T1600	311	10
s15850-T100	2,420	26
s35932-T100	6,408	14
s35932-T200	6,405	12
s35932-T300	6,405	37
s38417-T100	5,799	11
s38417-T200	5,802	11
s38417-T300	5,801	44
s38584-T100	7,343	19
s38584-T200	7,374	96
s38584-T300	7,615	873

学習フローでは，まず学習対象のベンチマークから，表 1 に示した 1–11 の 11 個の特徴量を抽出する (Step L1.1)．抽出した 11 個の特徴量を用いて，NN で学習対象のベンチマークを学習，分類器を生成し，ハイパーパラメータを調整する (Step L1.2)．生成した学習済みの分類器を用いて，学習対象のベンチマークに含まれるネットをノーマルネットとトロイネットに分類し，Trojan_prob を取得する (Step L1.3)．Step L1.3 で取得した Trpjan_prob をもとに，学習対象のベンチマークから特徴量 Trojan_0，Trojan_in_1，Trojan_out_1 を抽出し，Step L1.1 で取得した 11 個の特徴量と合わせて 15 個の特徴量とする (Step L2.1)．抽出した 15 個の特徴量を用いて，学習対象のベンチマークを学習，分類器を生成し，ハイパーパラメータを調整する (Step L2.2)．

識別フローでは，識別対象のベンチマークから，表 1 に示した 1–11 の 11 個の特徴量を抽出する (Step C1.1)．Step L1.2 で生成した学習済みの分類器を用いて，11 個の特徴量で識別対象のベンチマークに含まれるネットをノーマルネットとトロイネットに分類し，Trojan_prob を取得する (Step C1.2)．Step C1.2 で取得した Trpjan_prob をもとに，識別対象のベンチマークから，特徴量 Trojan_0，Trojan_in_1，Trojan_out_1 を抽出し，Step C1.1 で取得した 11 個の特徴量と合わせて 15 個の特徴量とする (Step C2.1)．Step L2.2 で生成した学習済みの分類器を用いて，15 個の特徴量で識別対象のベンチマークに含まれるネットをノーマルネットとトロイネットに分類する．(Step C2.2)．

ハードウェアトロイ識別において適用する NN の構造を図 6 に示す．NN は入力層，中間層，出力層により構成される．特徴量の数が，第一段階の識別では 11 個，第二段階の識別では 3 章で提案した 4 つの特徴量を考慮した 15 個であるため，入力層のユニット数はそれぞれ 11 と 15 とした．出力層のユニット数は，各ネットをトロイネットまたはノーマルネットの 2 つに分類するため 2 とした．中間層は複数段から構成されることがあり，本稿では文献 [5] で最良の結果を得た中間層 3 段の構成とし，それぞれのユニット数を 200，100，50 の 3 段とした．

5. 評価実験

5.1 実験結果

本章では提案手法の実験結果を報告する．実験にはメモリ 1056GB，CPU として Intel Xeon E7-8855 v4 を搭載するコンピュータを用いた．図 5 のアルゴリムは Python 3.5 のライブラリ Chainer [12] を使用し実装した．NN には勾配降下法として Adam，活性化関数として tanh を使用した．Trust-HUB [13] で公開されている 17 個のベンチマークを学習と識別に用い，交差検証で評価した．表 2 に学習と識別に用いた 17 個のベンチマークを示す．

表 3 に第一段階の識別結果 (Step C1.2) と第二段階の識別結果 (Step C2.2) の比較を示す．表 3 に示すトロイネット割合は，第一段階の識別でトロイネットと分類されたネットが全ネットに対して占める割合を表す．第一段階の識別の結果は，平均 TPR 79.3%，平均 TNR 97.2%，を示し，第二段階の識別の結果は，平均 TPR 80.5%，平均 TNR 96.9%，を示した．第二段階の識別は第一段階の識別に比べ，平均 TPR は 1.21 ポイント向上し，平均 TNR は 0.30 ポイント低下した．17 個中 12 個のベンチマークで TPR は変化なし，あるいは向上し，12 個のベンチマークで TNR は向上した．

5.2 トロイネット割合に着目したハードウェアトロイ識別

表 3 において，TPR が第一段階の識別に比べ第二段階の識別で 8 ポイント以上減少したベンチマークは全部で 4 個あり，その全てがトロイネット割合が 6.0%未満であっ

Vol.2019-ARC-235 No.51
Vol.2019-SLDM-187 No.51
Vol.2019-EMB-50 No.51
2019/3/18

表 3: 第一段階の識別結果と第二段階の識別結果の比較.

ベンチマーク	第一段階の識別		第二段階の識別		二段階の識別の比較		トロイネット割合 [%]
	TPR [%] (i)	TNR [%] (ii)	TPR [%] (iii)	TNR [%] (iv)	TPR 増減 (iii) − (i)	TNR 増減 (iv) − (ii)	
RS232-T1000	100.0	95.1	100.0	95.5	0.0	0.4	7.84
RS232-T1100	100.0	96.8	100.0	98.1	0.0	1.3	6.56
RS232-T1200	100.0	95.5	92.3	97.1	-7.7	1.6	8.36
RS232-T1300	85.7	96.8	85.7	97.7	0.0	0.9	5.06
RS232-T1400	100.0	97.1	100.0	92.8	0.0	-4.3	6.60
RS232-T1500	100.0	95.8	100.0	96.5	0.0	0.7	7.45
RS232-T1600	80.0	94.5	100.0	98.1	20.0	3.6	7.79
s15850-T100	80.8	95.6	61.5	97.5	-19.2	1.9	5.19
s35932-T100	64.3	100	85.7	99.5	21.4	-0.5	0.56
s35932-T200	16.7	99.4	8.33	99.7	-8.3	0.3	0.65
s35932-T300	89.2	100	97.3	99.6	8.1	-0.4	0.85
s38417-T100	100.0	98.0	63.6	99.4	-36.4	1.4	2.15
s38417-T200	100.0	97.4	81.8	99.1	-18.2	1.7	2.80
s38417-T300	97.7	98.5	97.7	98.7	0.00	0.2	2.22
s38584-T100	15.8	99.1	15.8	99.4	0.00	0.3	0.94
s38584-T200	62.5	94.3	90.6	90.8	28.1	-3.4	6.45
s38584-T300	55.1	98.6	87.9	87.8	32.8	-10.9	6.88
平均値	79.3	97.2	80.5	96.9	1.2	-0.3	4.61

表 4: トロイネット割合を考慮した識別結果

ベンチマーク名	TPR [%]	TNR [%]
RS232-T1000	100.0	95.5
RS232-T1100	100.0	98.1
RS232-T1200	92.3	97.1
RS232-T1300	85.7	96.8
RS232-T1400	100.0	92.8
RS232-T1500	100.0	96.5
RS232-T1600	100.0	98.1
s15850-T100	80.8	95.6
s35932-T100	64.3	100
s35932-T200	16.7	99.4
s35932-T300	89.2	100
s38417-T100	100.0	98.0
s38417-T200	100.0	97.4
s38417-T300	97.7	98.5
s38584-T100	15.8	99.1
s38584-T100	90.6	90.8
s38584-T100	87.9	87.8
平均値	83.6	96.5

た．提案手法は，ハードウェアトロイの局所性を考慮しトロイネットに分類されたネットをもとに他のトロイネットを発見する手法であるため，トロイネット割合が小さい

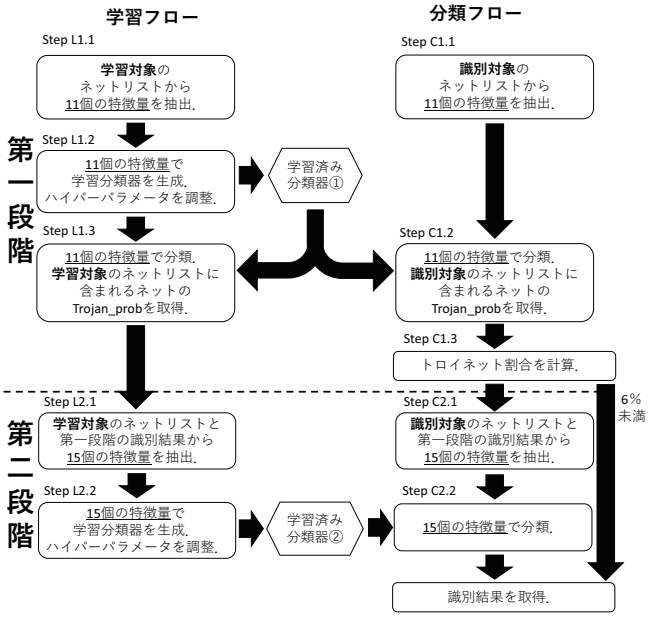

図 7: トロイネット割合を考慮した提案手法の流れ.

ハードウェアトロイはトロイネットに関する周辺情報が少なく，第二段階の識別で TPR が向上しない．そこで，トロイネット割合が 6.0%未満のベンチマークは第一段階の識別結果を最終的な識別結果にする．

以上の議論により，図 5 における Step C1.2 の次に以下の手順を追加する．

Vol.2019-ARC-235 No.51
Vol.2019-SLDM-187 No.51
Vol.2019-EMB-50 No.51
2019/3/18

Step C1.3 トロイネット割合が 6.0%未満のベンチマークは第一段階の識別結果を提案手法の識別結果とする．

図 7 に上記の Step C1.3 を追加した提案手法の流れを示す．表 4 にトロイネット割合を考慮した識別結果を示す．トロイネット割合に郷里した識別結果は平均 TPR 83.6%，平均 TNR 96.5%となった．NN を用いた検出手法 [6] を適用した第一段階の識別結果と比較すると，平均 TPR は 4.3 ポイント向上し，TNR は 0.7 ポイント低下にとどまっている．

以上の実験結果から，第一段階の識別結果がネットの誤分類を含んでいたとしても，その識別結果をもとづく特徴量を第二段階の識別で利用することで，全体の識別性能は向上する．

6. おわりに

本稿ではハードウェアトロイの局所性をもとに周辺ネットの特徴量を考慮したハードウェアトロイ検出手法を提案した．提案手法は識別対象のネットリストを 11 個の特徴量で分類し，新たに 4 個の特徴量を取得する．合計 15 個の特徴量で識別対象のネットリストを再度識別する．Trust-HUB [13] で公開されている 17 個のベンチマークに対し，交差検証を適用し評価実験した．評価実験の結果，トロイネット割合が 10%以下のベンチマークにおいて TPR の大きな低下が見られたため，トロイネット割合が 10%以下のベンチマークは第一段階の識別結果を提案手法における識別結果とした．提案手法における識別結果は，平均 TPR が 83.6%，平均 TNR が 96.5%を示した．第二段階の識別は，NN を用いたハードウェアトロイ検出手法 [5] を適用した第一段階の識別に比べ，平均 TPR は 4.3 ポイント向上し，同等の平均 TNR を示した．

今後はより高い TPR，TNR を目指し，特徴量や手法の手順を改良する．また，提案手法をネットリストに適用する前に，ネットリストにハードウェアトロイが挿入されているかどうかを判別する手法について考察する予定である．

謝辞 本研究の一部は，総合科学技術・イノベーション会議の戦略的イノベーション創造プログラム (SIP) 第 2 期「IoT 社会に対応したサイバー・フィジカル・セキュリティ」（管理法人：NEDO）によって実施されている．

参考文献

[1] A. Bazzazi, M. T. Manzuri Shalmani, and A. M. A. Hemmatyar, “Hardware Trojan detection based on logical testing,” *Journal of Electronic Testing*, vol. 33, no. 4, pp. 381–395, 2017.

[2] N. Cornell and K. Nepal, “Combinational hardware Trojan detection using logic implications,” in *IEEE International Midwest Symposium on Circuits and Systems (MWSCAS)*, 2017, pp. 571–574.

[3] J. Francq and F. Frick, “Introduction to hardware trojan detection methods,” in *Proc. Design, Automation and Test in Europe (DATE)*, 2015, pp. 770–775.

[4] K. Hasegawa, M. Oya, M. Yanagisawa, and N. Togawa, “Hardware Trojans classification for gate-level netlists based on machine learning,” in *Proc. IEEE Symposium on On-Line Testing and Robust System Design (IOLTS)*, 2016, pp. 203–206.

[5] K. Hasegawa, M. Yanagisawa, and N. Togawa, “A hardware-Trojan classification method using machine learning at gate-level netlists based on Trojan features,” *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E100.A, no. 7, pp. 1427–1438, 2017.

[6] ——, “Hardware Trojans classification for gate-level netlists using multi-layer neural networks,” in *Proc. IEEE Symposium on On-Line Testing and Robust System Design (IOLTS)*, 2017, pp. 227–232.

[7] J. He, Y. Zhao, X. Guo, and Y. Jin, “Hardware Trojan detection through chip-free electromagnetic side-vhannel statistical analysis,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 10, pp. 2939–2948, 2017.

[8] D. Jap, W. He, and S. Bhasin, “Supervised and unsupervised machine learning for side-channel based Trojan detection,” in *Proc. International Conference on Application-Specific Systems, Architectures and Processors*, 2016, pp. 17–24.

[9] Jun Li, Lin Ni, Jihua Chen, and E. Zhou, “A novel hardware Trojan detection based on BP neural network,” in *Proc. International Conference on Computer and Communications (ICCC)*, 2016, pp. 2790–2794.

[10] M. Oya, Y. Shi, M. Yanagisawa, and N. Togawa, “A score-based classification method for identifying hardware-trojans at gate-level netlists,” in *Proc. Design, Automation and Test in Europe (DATE)*, 2015, pp. 465–470.

[11] H. Salmani, “COTD: Reference-free hardware Trojan detection and recovery based on controllability and observability in gate-level netlist,” *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 2, pp. 338–350, 2017.

[12] S. Tokui, K. Oono, S. Hido, and J. Clayton, “Chainer: a next-generation open source framework for deep learning,” in *Proc. Workshop on Machine Learning Systems (LearningSys) in The Twenty-ninth Annual Conference on Neural Information Processing Systems (NIPS)*, 2015.

[13] “Trust-HUB.” http://www.trust-hub.org