Vol.2021-CSEC-94 No.1
Vol.2021-SPT-43 No.1
2021/7/19

マルウェア検知に対するバックドアポイズニング攻撃の対策としてのオートエンコーダの定量的評価

松本 悠希[1,a)]　成定 真太郎[2,b)]　披田野 清良[2,c)]　内林 俊洋[3,d)]　菅沼 拓夫[1,e)]　樋地 正浩[1,f)]

概要：マルウェア解析では，亜種の増加により解析者の負担が増大しており，効率的にマルウェアを検知可能な機械学習による静的解析の需要が高まっている．一方，機械学習に対する代表的な脅威として，バックドアポイズニング攻撃がある．本攻撃では，訓練データにポイズニングデータを混入することで，特定のデータを誤分類させる．Severi らの報告により，本攻撃は，機械学習を用いたマルウェア検知へも応用できることが明らかとなっている．このため，機械学習によりマルウェアを高精度に検知するためには，ポイズニングデータの影響を受けない攻撃耐性の高い検知モデルの構築が必要となる．本稿では，検知モデルを構築する際にポイズニングデータの含まれていないクリーンなデータを入手できない状況を想定し，ポイズニングデータ入りの訓練データに対してオートエンコーダを適用することでポイズニングデータの効果を排除する方法を提案する．そして，実データを用いた評価実験を通して，提案手法により，マルウェアの検知精度の低下を最小限に抑えつつ，バックドア攻撃の影響を大幅に低減できることを示す．

キーワード：バックドアポイズニング攻撃，マルウェア検出，オートエンコーダ．

1. はじめに

多くの企業がサイバー攻撃の脅威にさらされており，そのセキュリティ対策が必須となっている．その中でもマルウェアの存在はインターネットに繋がるあらゆる端末に対して悪影響を及ぼす可能性があることから，早い段階で検知，除去することが重要となる．ここで，マルウェア解析手法には大きく分けて静的解析と動的解析の 2 種類が存在する．静的解析は，ファイルのバイナリ自体を解析して特徴量を抽出し，マルウェアの判別を行う手法である．その代表的な手法の一つにパターンマッチング [1][2] がある．これはマルウェアに特有の特徴量を蓄積しておき，判定対象のファイルの特徴量と比較することによりマルウェアを検出する方法であり，既知のマルウェアを低コストで確実に検出できる．動的解析 [3] は，仮想環境等で実際にプログラムを動作させることでマルウェアかどうかを判断する手法であり，未知のマルウェアに対しても検出できる反面，検出のためのコストが高い．近年，マルウェア亜種の生成が容易になったことで，静的解析によるマルウェアの検出が困難になりつつある [4]．マルウェア亜種とは，既知のマルウェアの一部をわずかに改変することで作られるマルウェアであり，基本的な動作はオリジナルのマルウェアと変わらないものの，特徴量が異なることからパターンマッチングでは検出できないことが確認されている．一方，動的解析では検出が可能であるが，検出のコストが高いことから増え続けるマルウェア亜種に対応するのは非常に困難である．そこで最近では機械学習による高精度な静的解析手法が注目されている．

その一方で機械学習の脆弱性も発見されている．敵対的サンプルがその 1 つであり，テストデータに小さなノイズを加えるだけでモデルの誤分類を引き起こすことが可能となる．また，訓練データにノイズが加えられたデータを混入させることで，特定のデータのみの誤分類を引き起こすバックドアポイズニング攻撃 [5] も確認されている．特に，Severi ら [6] によって提案されたバックドアポイズニング攻撃では，特定のマルウェアのみをマルウェアではないと誤分類させるためのポイズニングデータを解析前のバイナリファイルのデータセットの中に混入することができるた

[1] 東北大学
Tohoku University, Sendai, Japan
[2] KDDI 総合研究所
KDDI Research, Inc., Fujimino, Japan
[3] 九州大学
Kyushu University, Fukuoka, Japan
[a)] matsumoto@ci.cc.tohoku.ac.jp
[b)] sh-narisada@kddi-research.jp
[c)] se-hidano@kddi-research.jp
[d)] uchibayashi.toshihiro.143@m.kyushu-u.ac.jp
[e)] suganuma@tohoku.ac.jp
[f)] hiji@tohoku.ac.jp

め，攻撃者は防御者がデータを収集する時点でポイズニングデータを仕掛けることが可能となる．そのため，防御側はポイズニングデータが混入した訓練データを利用することを想定する必要があるが，私たちの知る限りでは，この攻撃に対する有効な対策はない．

本論文では Severi らが提案したマルウェア分類器に対するバックドアポイズニング攻撃に対する効果的な対策として，訓練データをオートエンコーダに通すことでバックドアを除去する方法を提案し，訓練データの特徴量を単純に削減する方法と比較した．その結果，Severi らが提案した 2 つのバックドアポイズニング攻撃の攻撃成功率を，マルウェアの分類精度を低下させることなく，大きく低減させることができた．

2. 先行研究

2.1 マルウェア分類器に対する脅威

静的解析に基づく機械学習を用いた一般的なマルウェア分類に使用されるデータの収集過程について説明する．まず，悪意のあるソフトウェアや通常のソフトウェアのバイナリファイルが脅威情報プラットフォームに収集される．これらのバイナリファイルは，オンライン上のあらゆる空間から収集されるが，攻撃者を含むユーザーが提供することもある．収集されたバイナリファイルは，既存のアンチウイルスエンジンを用いた解析により，自動的にラベル付けされる．こうして収集されたデータを特定の企業や研究者がデータセットとして使用することができる．ここで，収集されるデータは膨大な数となることから，すべてを正確に区別することは困難であり，攻撃者の介入も予想されることから，一定の割合でポイズニングデータが混入することが考えられる．このようにポイズニングデータが混入した状態でデータセットが生成され，機械学習モデルの訓練が行われた場合，生成されたモデルが悪意のあるソフトウェアを害のないソフトウェア（グッドウェア）として誤分類することが起こりえる．

このような中でマルウェア分類器に対する攻撃や防御に関するいくつかの研究がある．

2.2 攻撃手法

Severi ら [6] は SHAP（SHarpley Additive exPlanations）[7] に基づく攻撃アルゴリズムを提案している．SHAP は，学習済みのモデルにおいて，ある特徴量がそのモデルの予測値に対してどれだけ寄与したかを算出するものである．それにより，ある特徴量のマルウェア分類における貢献度を可視化することが可能となる．具体的には，ある入力 v と学習済みモデル f が与えられた時，予測結果 $f(v)$ を各特徴量の寄与度が説明しやすい簡易的なモデル g により近似される．g は以下の式により表される．

$$g(z) = \phi_0 + \sum_{i=1}^{p} \phi_i z_i \tag{1}$$

ここで，$z_i \in \{0,1\}$，$\phi_i \in R$ である．z_i は各入力 v を単純化したものであり，i 番目の特徴量が予測に寄与していれば 1，予測に寄与していなければ 0 を取る．また，ϕ_i の値において $\phi_i > 0$ の場合はマルウェア，$\phi_i < 0$ の場合はグッドウェアの方向に寄与しており，それらを合計することで元の予測結果 $f(v)$ の近似値を得る．実験では，表 1 に示す 2351 次元から構成される EMBER データ [8] のうち，SHAP 値に従って一部の次元を操作することでポイズニングデータを生成し，訓練データ全体の 1 %のポイズニングデータの注入により分類モデルを汚染する事でバックドアポイズニング攻撃を行っている．ここで，Severi らは 2 つのバックドア生成アルゴリズムを使用することで攻撃対象の次元を選択し，攻撃成功率の高いバックドアポイズニング攻撃を実現している．1 つは Independent Selection と呼ばれる手法であり，これは EMBER データの特徴量の中でも SHAP 値の絶対値の和が最大の次元を選択し，さらに選択された次元で SHAP 値の出現頻度が最小となる値をバックドアとして選択することで，決定境界に強い影響を与えることができる．もう 1 つは Greedy Combined Selection と呼ばれる手法であり，EMBER データの特徴量の中でも SHAP 値の和が最小となる次元を選択し，さらに選択された次元で SHAP 値の和が負の値かつ出現頻度が多い値をバックドアとすることで，グッドウェアの領域を大きく破壊することができる．実際に攻撃者がバイナリファイルを操作する際は，マルウェアとして動作させるために，慎重な操作を行わなければならない．EMBER データの次元の大部分（2351 次元のうち 2316 次元）は，ハッシュに基づく値を持つ次元であり，その値を操作することは困難である．操作可能な次元は，非ハッシュ値を持つ 35 の次元のみであることが確認されており，それらの次元はすべて 513 次元から 2351 次元の間に存在している．

より強いバックドアポイズニング攻撃の一例にラベルフリップ [9] 攻撃がある．これは，攻撃者がデータに加えてデータに付与されたラベルを操作する能力を持っている場合に訓練データに誤ったラベルを付与することで，モデルの予測精度を低下させる攻撃である．敵対的なデータの生成手法である FGSM（Fast Gradient Sign Method）[10] と組み合わせることで，マルウェアをグッドウェアとして誤分類させるより強力なバックドアポイズニング攻撃が可能となる．

2.3 防御手法

HDBSCAN[11] と呼ばれる手法は，密度ベースのクラスタリング手法を発展させたものであり，密度の閾値を無限に変化させることで柔軟なクラスタリングが可能であると

表 1: EMBER データの構造

No.	次元	Feature Group	概要
1	1-256	ByteHistgram	各バイト (0-255) の出現回数
2	257-512	ByteEntropyHistgram	各バイトとエントロピーの同時分布
3	513–767	SectionFileInfo	各セクションのサイズ等の情報
4	768–2047	ImportsInfo	外部ライブラリ等の情報
5	2048–2175	ExportsInfo	外部関数のリスト
6	2176–2185	GeneralFileInfo	ファイルサイズといったファイルの概要
7	2186–2247	HeaderFileInfo	バージョン等のヘッダ情報
8	2248–2351	StringExtractor	ファイルに含まれる文字数等の情報

いう利点を持つ．また，別の手法 [12] では，バックドア攻撃が行われた入力には検出可能な痕跡（Spectral signatures）が残ることを利用して，攻撃により操作された入力を識別・除去できることが確認されている．しかし，モデルの学習にはポイズニングデータを含まないクリーンなデータを用いることが前提である．一般的にはマルウェア分類のために使用する訓練データはオープンソースの情報共有プラットフォームから収集されるが，それらのデータは悪意のある攻撃者を含むユーザーから提供されているため，一定のポイズニングデータが混入する可能性がある．従って，クリーンなデータから学習を行うことは現実的ではない．

Isolation Forest[13] は，異常検知あるいは外れ値検知を行うための教師なし学習法であり，バックドア検知にも使用できる．実際には，ランダムに特徴量を選択したうえでランダムに区切り値を選択し，それによりデータを分割していくことで決定木を形成していく．ここで，異常な値は他の値から簡単に分離できるという特性を持つため，分割する回数が少ない程異常値としてみなすことが可能になる．ただし，この手法は Severi らの攻撃アルゴリズムに対して有効でないことが示されている．

Lee ら [14] は，次元削減や特徴抽出に使用されるオートエンコーダを防御アルゴリズムとして採用している．攻撃者は敵対的なデータを生成して対象となるデータセットに混入させるが，敵対的なデータは人間には発見することが非常に困難であるため，簡単に検知を回避してしまう．そこで，分類モデルが汚染されたデータセットで学習する前に，データセットを生成モデルとしても使用できるオートエンコーダを使って疑似データに置き換えることで，敵対的なデータを無効化した上で正常な学習を行うことを可能としている．しかし，モデルの学習にはクリーンなデータの使用を前提としている．また，テスト時にはデータの前処理を行っていないことから想定外の敵対的サンプルを防げない可能性がある．

3. 提案

2 章で述べたように，分類モデルの学習に用いる膨大な訓練データの収集は，一般的にはオープンソースの情報共有プラットフォームから収集されるため，ポイズニングデータの混入がないと考えるのは非現実的である．そのため，ポイズニングデータの混入を想定した上でモデルの学習を行うことを前提に考える必要がある．また，仮にモデルの学習段階でポイズニングデータを無効化できたとしても，テストの段階で想定外の敵対的サンプルが混入した場合攻撃を防げない可能性がある．したがって，モデルの学習とテストの段階の両方で包括的な対策を行う必要がある．

ここで，防御者が EMBER によって特徴量の抽出を実施している場合，攻撃者がバイナリファイル上において任意の値に操作できる特徴量空間の次元は表 1 のうち 513 次元から 2351 次元であること [6] が確認されている．したがって，次に説明する 2 つの提案手法により毒の除去を試みる．

3.1 次元削減を用いた対策手法

大山ら [15] は，EMBER データセット [8] の 2351 次元の特徴量から一部の特徴量のみを使用することで LightGBM を用いた分類モデルの学習の時間を抑えつつ一定の精度を保つことに成功している．そこで，攻撃者が毒を加えると想定される 513 次元から 2351 次元を削除することで理論上バックドアによる攻撃成功率を 0 にすることができると考えられる．ただし，モデルの学習に使う特徴量を一部削除することによる一定の検知漏れが起こることが確認されているため，より高精度な検知手法が必要となる．

3.2 オートエンコーダを用いた対策手法

次元削減による毒の除去では，重要な特徴量が削除されることで分類モデルの精度低下が起こることから，特徴量の削減を行わない手法としてオートエンコーダを使用した疑似データへの置換による毒の除去を提案する．図 1 にオートエンコーダによる対策手法の概要を示す．オートエンコーダは，n 個の入力を m 個の出力まで圧縮するニューラルネットワークであるエンコーダと，m 個の入力を n 個の出力まで復元するニューラルネットワークであるデコーダの 2 つのニューラルネットワークから構成される（$n > m$）．2 章の先行研究 [14] でも述べたように，入力データをオートエンコーダによって圧縮・復元処理を行う

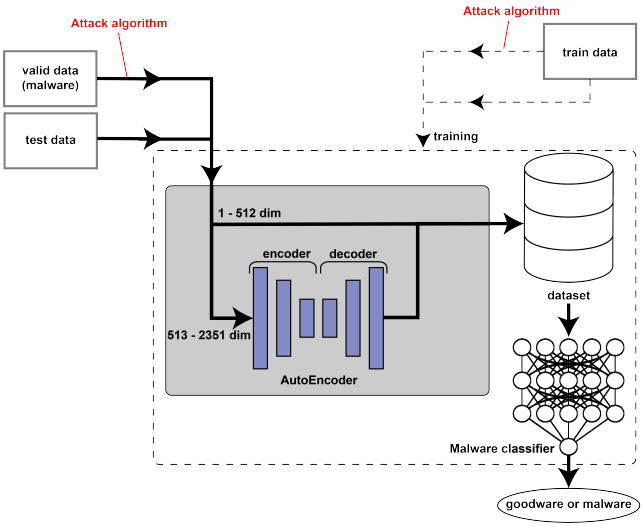


図 1: オートエンコーダを用いた提案手法の概要

ことで，攻撃者が挿入する毒の除去が可能である．従って，攻撃者が毒を加えると想定される 513 次元から 2351 次元を，オートエンコーダによって処理することで疑似データに置き換え，最初の 512 次元と組み合わせることにより精度の高い分類モデルの構築を提案する．この手法により，分類モデルのポイズニングデータによる汚染を最小限に抑えることが可能となる．また，オートエンコーダの学習パラメータ θ_{AE} は以下の式により最適化を実行している．

$$\theta_{AE} =: \arg\min_{\theta_{AE}} \frac{1}{|\hat{D}_{tr}|} \sum_{(s,x)\in\hat{D}_{tr}} ||Dec(Enc(x))-x||_2^2 (2)$$

$\hat{D}_{tr}$ はポイズニングデータを含む訓練データセット，s は攻撃者が操作不可能な次元，x は攻撃者が操作可能な次元であり，元のデータは (s,x) と表される．入力 x の次元はエンコーダである $Enc(x)$ により圧縮され，デコーダである $x_0 = Dec(enc(x))$ により復元されることで，データセットは疑似データである (s,x_0) に置き換えられる．ここで，我々の手法では先行研究とは違い，オートエンコーダの学習の時点で毒の混入を許容することでより現実的な分類モデルの構築を行っている．また，モデルを訓練する時に用いる訓練データだけではなくテストデータと検証データもオートエンコーダによる処理を行うことで想定外の敵対的サンプルによる攻撃の対策も行っている．なお，エンコーダとデコーダは訓練時に構築したものをそのまま利用することとする．

4. 実験

実験に使用するデータは，EMBER データセットからグッドウェアとマルウェアを 1：1 の割合で合計 1 万個になるようランダムに抽出し，8：2 の割合で訓練データとテストデータに分割した．また，マルウェアがグッドウェアに誤分類される割合である攻撃成功率を確認するための検証データは，EMBER データセットの中からマルウェアをランダムに 1000 個選択することにより生成した．ここで，EMBER データセットは 2351 次元の特徴量から構成されており，そのうち攻撃者が操作可能な次元は 513 次元から 2351 次元までであること [6] が分かっている．

攻撃側は 2351 次元のうち一部の特徴量を操作したポイズニングデータを元の訓練データに一定の割合で混入させることで汚染させた分類モデルを生成し，検証データにはすべて毒を付与することでバックドアポイズニング攻撃を再現し，評価する．ここで，攻撃手法は Severi ら [6] が提案した Independent Selection と Greedy Combined Selection を使用する．また，実際にはラベルの付与は攻撃者ではなく既存のアンチウイルスエンジンによって行われることから，ラベルを意図的に反転させる攻撃は非実用的であるが，防御側の視点からより強力なバックドアポイズニング攻撃に対して提案手法が有効であるかを調べるために，ラベルの反転を行った FGSM に基づく攻撃についても評価する．

マルウェア分類器には，先行研究 [6] の EmberNN と同様のニューラルネットワークモデルを実装した．このニューラルネットワークは 4 層の密結合された線形層から構成されており，最初の 3 層は入力と同様のノード数としている．活性化関数には最初の 3 層には ReLU 関数を，最後の層には Sigmoid 関数を使用した．また，予測精度を向上させるために線形層の間にはバッチ正規化とドロップアウトを適用した．そして，データをマルウェア分類器に通す前に EMBER データの 513 次元から 2351 次元をオートエンコーダによって前処理を行う．オートエンコーダは 2 層のエンコーダとデコーダで構成し，過学習を防ぐためにエンコーダの最後の層の後にドロップアウトを適用している．オートエンコーダによる圧縮は 1839 次元から 512 次元である．それに加え，オートエンコーダのエンコーダのみの圧縮操作と，デコーダによる復元処理を含む操作において毒の除去性能や精度に差があるのかを調べるため，先頭 512 次元とエンコーダにより 513 次元から 2351 次元を 512 次元に圧縮したデータを組み合わせた場合についても検証を行う．生成された疑似データの次元は 1024 次元となるが，その際マルウェア分類器の構造は EmberNN と同様のニューラルネットワークモデルとしており，最初の 3 層を入力と同様のノード数としている．また，次元削減に関しては EMBER データのうち 513 次元から 2351 次元を削除した先頭 512 次元のみを使用し，マルウェア分類器に通すことで検証を行う．こちらに関してもマルウェア分類器は上記 2 つと同様の構造とする．なお，テスト時にテストデータ及び検証データをオートエンコーダにより前処理することで，データをそのまま使用する場合よりも攻撃成功率を低減できることが確認されている．

5. 結果

3 種類の攻撃に対する防御手法の性能を測定した．その

結果を図に示す．ここで，性能については精度（$Accuracy$）と攻撃成功率（$AttackSuccessrate$）により測定している．式は表 2 に記載されている評価指標を使用すると以下の通りである．

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{3}$$

$$AttackSuccessrate = \frac{FN}{FN + TN} \tag{4}$$

表 2: 評価指標

		正解	
		グッドウェア	マルウェア
予測	グッドウェア	TP	FP
	マルウェア	FN	TN

評価については，バックドア混入後における対策なしの場合（No Defence），次元削減を行った場合（Deletion），オートエンコーダのうちエンコーダのみを使用した場合（AE_enc），そしてオートエンコーダを使用した場合（AE）について計測を行い，訓練データにおけるポイズニングデータの混入割合を $p = 1(\%)$，2351 次元のうち毒が混入した次元数を $N = 8$ に固定し，徐々に変化させることで検証を行った．

5.1 Independent Selection に対する防御性能

攻撃成功率の結果を図 2a と 2b に示す．対策なしの場合には攻撃成功率は N または p が大きくなるにつれて高くなっていき，100% に達する．ここで，$p = 0$ は訓練データにポイズニングデータに毒が混入していない場合であるが，これはバックドア攻撃ではなく回避攻撃による成功率と言い換えることができる．次元削減を行った場合には，攻撃成功率は N や p の値によって多少差はあるものの 0% から 20% まで減少させることができる．オートエンコーダを使用した場合には次元削減と同様の水準で攻撃成功率を減少させることに成功している．また，エンコーダによる次元圧縮のみを行った場合については，N や p の値が大きくなるにつれて攻撃成功率が高くなることから，デコーダによる復元処理が毒の除去性能に大きく寄与することが分かった．

精度に関する結果を図 2c と 2d に示す．対策なしの場合には N や p の値に関わらず 97% 前後を維持していることが確認できる．次元削減を行った場合，EMBER データの大部分を除去することから精度が大きく低下することが分かっており，約 90% 前後を推移している．一方オートエンコーダを使用した場合には次元削減と比較して約 3% 近くの改善がみられた．最後にエンコーダによる次元圧縮のみを行った場合と通常のオートエンコーダを使用した場合の比較では，前者の方がわずかに精度が高くなることが確認できた．

5.2 Greedy Combined Selection に対する防御性能

攻撃成功率の結果を図 3a と 3b に示す．対策なしの場合には攻撃成功率は Independent Selection よりも低く，最大でも 60% 近くまでの上昇に留まるが，これは先行研究と同様の結果となっている．次に次元削減やオートエンコーダで対策した場合に関しては，p や N の値に関わらず Independent Selection への対策時と同様の結果が得られた．

精度に関する結果を図 3c と 3d に示す．対策なしの場合には Independent Selection と同様に 97% 前後を維持していることが確認できる．次に次元削減やオートエンコーダで対策した場合に関しても，Indepent Selection と同様の結果が得られた．

5.3 FGSM（ラベルフリップあり）に対する防御性能

攻撃成功率の結果を図 4a と 4b に示す．対策なしの場合には上記 2 つの攻撃と比較して攻撃成功率が非常に高く，$N = 8$，$p = 1\%$ の時点で 100% 近くの攻撃が成功することが確認できる．次に次元削減を行った場合には，上記 2 つの攻撃を行った場合と同様に 20% 近くまで攻撃成功率を大きく低減できることが確認できた．一方，オートエンコーダを使用したところ，N や p の値が大きくなるにつれて攻撃成功率が上がってしまうことが確認できた．最後にエンコーダによる次元圧縮のみを行った場合には，オートエンコーダと比較して 20% から 40% ほど成功率が上昇してしまうことから，デコーダによる復元処理が防御性能の向上に大きく寄与することが判明した．

精度に関する結果を図 4c と 4d に示す．その結果，ラベルフリップによる攻撃に対しても精度に関しては上記 2 つの攻撃手法と同様の精度が維持されることが確認できた．

6. おわりに

本論文では Severi らが提案したマルウェア分類器に対するバックドアポイズニング攻撃に対する効果的な対策として，オートエンコーダを利用することで，疑似データへの置き換えによるデータ内のノイズの除去を，次元削減との比較により実験した．その結果，Severi らが提案した 2 つのバックドアポイズニング攻撃の攻撃成功率を，オートエンコーダが次元削減に匹敵する精度で大きく低減させることに成功した．また，オートエンコーダを使用することで，精度を大きく損なうことのない分類器の生成に成功した．それに加えて，オートエンコーダのデコーダによる復元処理が分類器の性能向上に大きく寄与することが確認できた．一方で，ラベルフリップが行われた場合については分類器の性能が大きく下がることが確認されたことから，今後の課題としては，ラベルフリップといった攻撃者にとっては非実用的であるものの強力な攻撃についても同時に解決できる手法の提案が必要である．

参考文献

[1] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, Vol. 86, No. 11, pp. 2278–2324, 1998.

[2] Tony Abou-Assaleh, Nick Cercone, Vlado Keselj, and Ray Sweidan. N-gram-based detection of new malicious code. In *Proceedings of the 28th Annual International Computer Software and Applications Conference, 2004. COMPSAC 2004.*, Vol. 2, pp. 41–42. IEEE, 2004.

[3] Konrad Rieck, Thorsten Holz, Carsten Willems, Patrick Düssel, and Pavel Laskov. Learning and classification of malware behavior. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, pp. 108–125. Springer, 2008.

[4] 中村燎太, 大山恵弘. ビヘイビアベースマルウェア検知におけるオンライン機械学習アルゴリズムの比較評価. コンピュータ ソフトウェア, Vol. 34, No. 4, pp. 4_156–4_177, 2017.

[5] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*, 2017.

[6] Giorgio Severi, Jim Meyer, Scott Coull, and Alina Oprea. Exploring backdoor poisoning attacks against malware classifiers. *arXiv preprint arXiv:2003.01031*, 2020.

[7] Scott Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *arXiv preprint arXiv:1705.07874*, 2017.

[8] Hyrum S Anderson and Phil Roth. Ember: an open dataset for training static pe malware machine learning models. *arXiv preprint arXiv:1804.04637*, 2018.

[9] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: identifying vulnerabilities in the machine learning model supply chain (2017). *arXiv preprint arXiv:1708.06733*, 2017.

[10] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

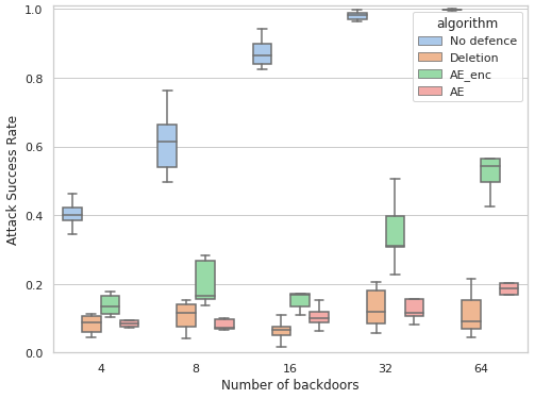
[11] Ricardo JGB Campello, Davoud Moulavi, and Jörg Sander. Density-based clustering based on hierarchical density estimates. In *Pacific-Asia conference on knowledge discovery and data mining*, pp. 160–172. Springer, 2013.

[12] Brandon Tran, Jerry Li, and Aleksander Madry. Spectral signatures in backdoor attacks. *arXiv preprint arXiv:1811.00636*, 2018.

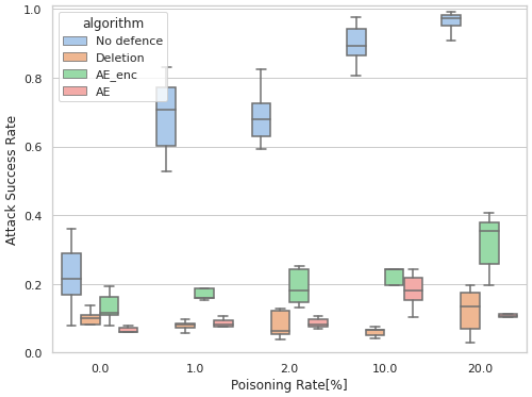
[13] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 eighth ieee international conference on data mining*, pp. 413–422. IEEE, 2008.

[14] Dongseop Lee, Hyunjin Kim, and Jaecheol Ryou. Poisoning attack on show and tell model and defense using autoencoder in electric factory. In *2020 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pp. 538–541. IEEE, 2020.

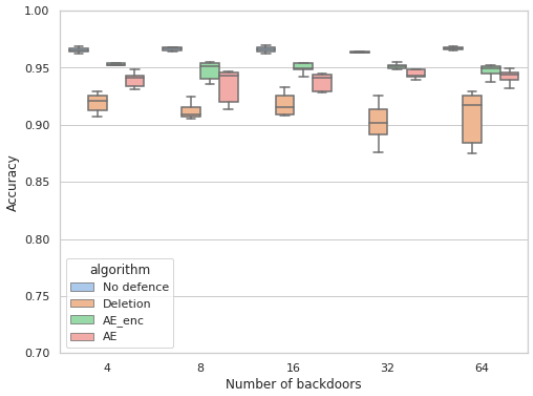
[15] Yoshihiro Oyama, Takumi Miyashita, and Hirotaka Kokubo. Identifying useful features for malware detection in the ember dataset. In *2019 seventh international symposium on computing and networking workshops (CANDARW)*, pp. 360–366. IEEE, 2019.



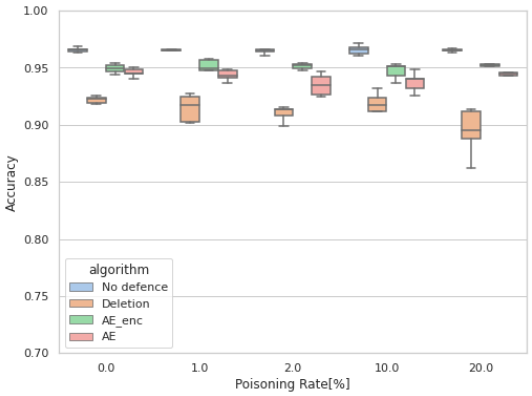
(a) N を変化させた時の攻撃成功率



(b) p を変化させた時の攻撃成功率

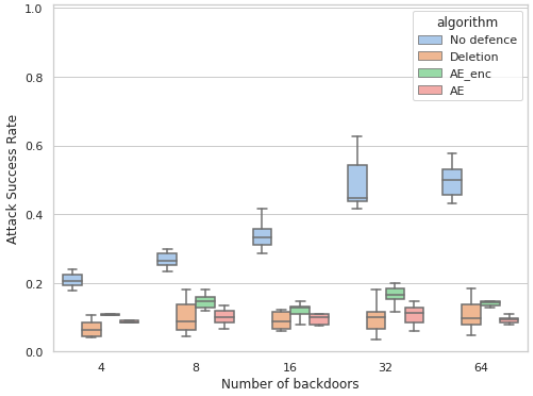


(c) N を変化させた時の精度

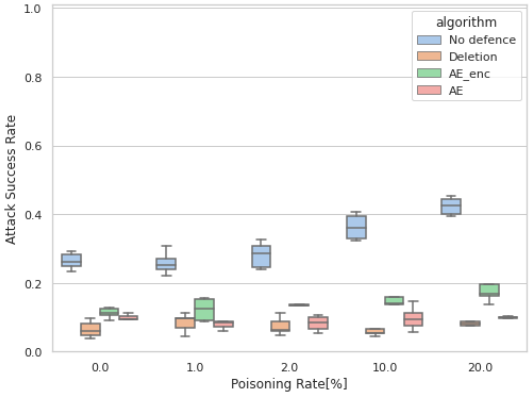


(d) p を変化させた時の精度

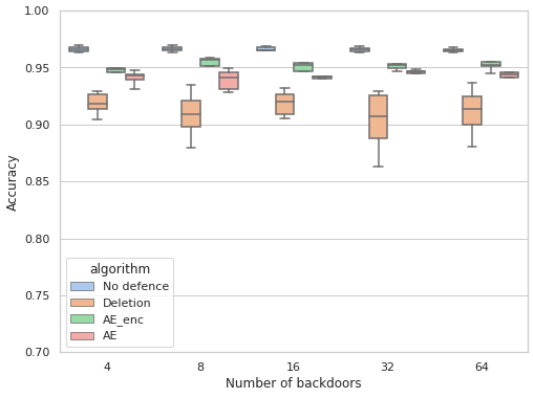
図 2: Independent Selection に対する防御性能



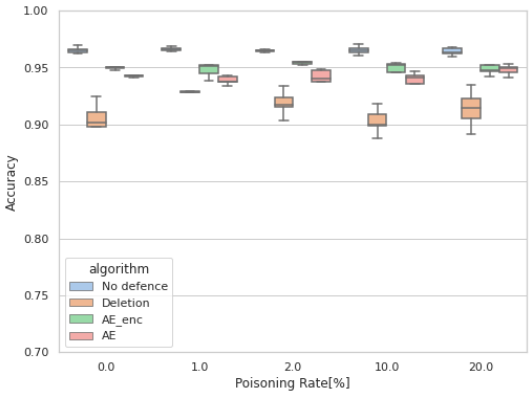
(a) N を変化させた時の攻撃成功率



(b) p を変化させた時の攻撃成功率

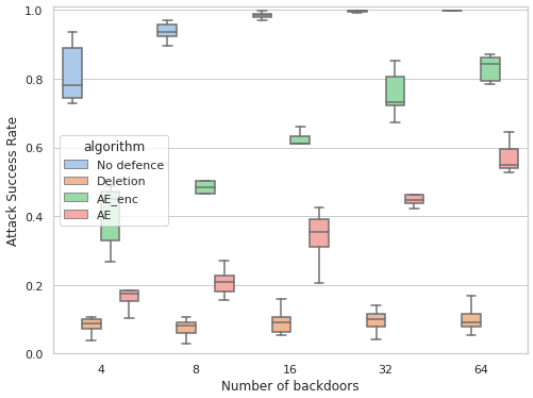


(c) N を変化させた時の精度

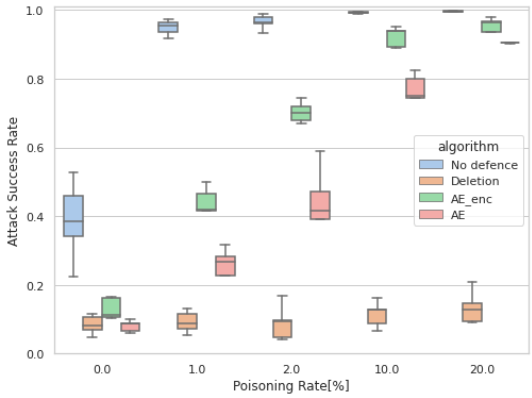


(d) p を変化させた時の精度

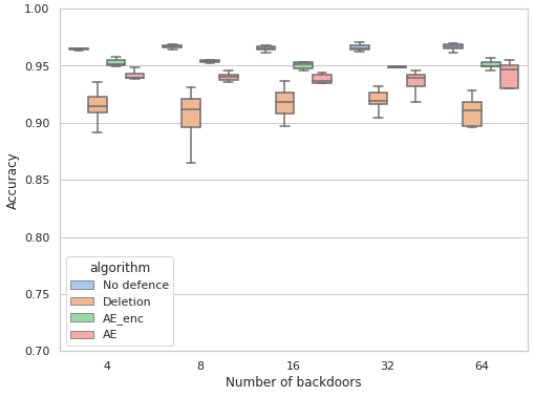
図 3: Greedy Combined Selection に対する防御性能



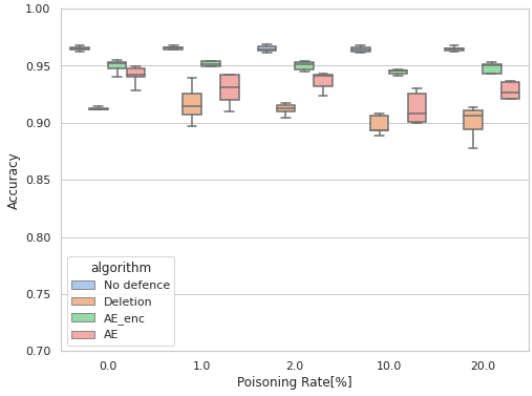
(a) N を変化させた時の攻撃成功率



(b) p を変化させた時の攻撃成功率



(c) N を変化させた時の精度



(d) p を変化させた時の精度

図 4: FGSM（ラベルフリップあり）に対する防御性能