Vol.2020-CSEC-91 No.22
Vol.2020-SPT-40 No.22
Vol.2020-EIP-90 No.22
2020/11/26

外部機構によるバックドアリングを用いた機械学習用電子透かし

四方 寿樹[1]　矢内 直人[1]　藤原 融[1]

概要：機械学習のモデルは一般に資産的価値が高いことから, 電子透かしの導入によるモデル自体の保護が注目されている. モデルと電子透かしの機構を分けることで特定の入力のみモデルを参照する方法（以降, 外部機構と呼ぶ）が近年提案された. しかし, 既存の外部機構方式では透かしを埋め込むパラメータの設計に関しての言及がなく, また, 内部機構として透かしを用いた場合との比較考察が不十分である. 本稿では, 外部機構を通じた電子透かし方式において, パラメータ生成に着目した検討を行う. これにより, 既存方式を含め外部機構における構成について, 実装と安全性の評価が詳細に行えるようになる. また, その提案方式を通じて外部機構の性質と内部機構に電子透かしを設けた方式の性質を比較考察することで, その特性を明らかにする. ResNet を用いて実験を行ったところ, 外部機構を持つ方式がモデル抽出攻撃および再学習への耐性を持つことを確認した.

キーワード：機械学習, 電子透かし, バックドア, 外部機構

1. はじめに

1.1 背景

機械学習モデルの構築において, 学習用データの収集と学習処理は一般に高い負荷を要する作業であり, その結果として得られるモデルはしばしば高い資産価値を有する. 例えば, Yang ら [20] のモデルでは学習に 61,000 ドルから 250,000 ドルも消費している[*1]. このため, コンテンツとしてモデルを保護することは, モデルの所有者における収益という観点では極めて重要な問題といえる. このような背景からコンテンツ保護技術として, 機械学習に対する電子透かしの適用 [17] が注目されており, 近年では研究が活発化している [1], [2], [21].

機械学習モデルへの電子透かしにおける近年の成果として, モデルの外部に電子透かしの生成機構を設ける方法 [15] が提案された. 従来方式 [1], [21] がモデルの内部に電子透かしを埋め込む「内部機構」と呼ばれることに対し, 文献 [15] の手法はモデルへのインターフェースとしてモデルと透かしの機構を明確に分けることから「外部機構」と呼ばれる. 外部機構の方式は, 従来の内部機構の方法 [1], [21] と比べ, 大まかには透かしを埋め込む処理の順番が異なる. 内部機構ではモデルの学習時に, 透かしとなる情報を学習データと併せて学習することで, モデルに埋め込む. 一方, 外部機構では学習済みのモデルに対して, 後から電子透かしを適用することで, モデル自体に作用することなく透かしを埋め込む.

外部機構の利点は保護対象となるモデルと保護機構そのものを切り分けることで, モデルそのものの更新に対する後方互換性を維持していることにある. 機械学習モデルはサービスの拡充などに向けて, 新たなデータを用いた再学習がモデルに対する更新処理としてしばしば行われる. このとき, 内部機構に基づく方式では, 再学習によりモデルのパラメータが更新されることで, 透かしが取り除かれる可能性がある. これに対し, 外部機構ではサーバ内のモデルと透かしが明確に分かれることで, 再学習による透かしへの影響を構造上取り除くこと, すなわち, モデルの柔軟な更新が期待できる. さらに, 機械学習の電子透かしにおける代表的な問題として, モデル内のパラメータを攻撃者が再学習により更新される再学習への耐性も知られているが, 外部機構はその潜在的な耐性も期待できる.

一方で, 外部機構の既存研究 [15] では透かしの埋め込みパラメータは検討がされていない. 内部機構ではモデル内のアーキテクチャが持つパラメータに対して透かしの埋め込みが保証される一方, 外部機構ではモデル内部の情報を使えないため, 埋め込みパラメータも詳細に設計する必要

[1] 大阪大学
大阪府吹田市山田丘 1-5

[*1] https://syncedreview.com/2019/06/27/the-staggering-cost-oftraining-sota-ai-models/

Vol.2020-CSEC-91 No.22
Vol.2020-SPT-40 No.22
Vol.2020-EIP-90 No.22
2020/11/26

がある. 関連して, 従来方式である内部機構に基づく方式と比較してどのような性能が得られるかも不明である.

1.2 貢献

本稿では外部機構を用いた構成として, (1) 電子透かしの埋め込みに関してパラメータを詳細に設計した方式を提案する. また, (2) ResNet を用いた実装を通じて, 外部機構と内部機構の性質の違いを実験的に評価する.

本稿の貢献は以下のとおりである. まず, 外部機構においてアルゴリズムが持つべきパラメータを初めて詳細に議論した. これにより, 既存方式含む外部機構方式の安全性を評価できるようになった. 関連して, 電子透かしの検証が成り立つ前提の下, 外部機構の要件及び安全性を計算論的に定式化しており, そのもとで具体的構成も示した. さらに, モデル抽出攻撃 [16] など透かしの検証という前提が成り立たない攻撃に対しても, 上述したパラメータ設計により, モデルの安全性を評価できるようになる.

次に, 具体的構成の実装を通じて, 内部機構に基づく既存方式 [1] と比較実験を行うことで, それぞれの構成における安全性を評価した. 具体的な攻撃手法としては攻撃者が透かしのないモデルを得る攻撃として, モデルの出力から攻撃者がモデルを複製するモデル抽出攻撃 [16] およびモデルから透かしを除去する再学習 [13] それぞれの攻撃耐性を明らかにしている. 結果として, 外部機構は上述した二つの攻撃に対して耐性を持つことを確認した.

2. 前提知識

本節では本稿に必要となる前提知識を述べる. とくに機械学習および機械学習へのバックドアについて, Adi ら [1] の記号計算モデルにおける定式化を示す.

2.1 機械学習

機械学習には学習データから最適なパラメータを持つモデルを得る学習処理と, そのモデルを利用してタスクを解決する推論処理がある. 以下に各処理に関する 計算論モデルでの定義 [1] を示す. これは学習データが何らかの正解情報（ラベル）を付与される教師あり学習となっている.

まず, 学習対象となるタスクの目的正解関数を f とする. ここで, f は与えられた入力を, あらかじめ定められたラベル集合のいずれかとして出力する関数である. このとき, 学習処理は f を近似的に計算できる関数 f' を探索する処理, 推論処理は学習済みの f' を利用する処理とみなせる. また, 学習の目標は未知のデータに対し高い精度で f と同じ出力を推論できる近似関数 f' の実現になる.

D を考えうる入力の集合, L は各入力に与えられるラベルの集合とし, それらの空間は任意の n に関する多項式 $p(n)$ で定義されるものとする. このとき上述した目的正解関数は $f : D \to L$ と書ける. この関数は, 人間が各入力に対しラベルを割り当てる作業に相当する. また, このとき正解ラベルを持つ入力集合を $\hat{D} = \{x \in D | f(x) = \perp\} \subseteq D$ とする. ここで, $\perp$ は未定義のラベルを意味する.

このとき, 機械学習は以下に示す学習 $Train$ と推論 $Infer$ の二つのアルゴリズムから定義される. まず, $Train$ はモデル $M \subset \{0,1\}^{p(n)}$, 学習データ $\boldsymbol{X} = \{x_1, ..., x_n\}$ とそのラベル $\boldsymbol{L} = \{l_1, ..., l_n\}$ を入力として与えられ, 新たなモデル M' を出力する確率的多項式時間アルゴリズムとする. また, $Infer$ は入力にモデル M, 推論対象のデータ $x \in D$ を与えられ, $M(x) \in L \backslash \{\perp\}$ を出力する確定的多項式時間アルゴリズムとする. 与えられた関数 f において, $\Pr[f(x) \neq Infer(Train(M, \boldsymbol{X}, \boldsymbol{L}), x) | x \in \hat{D}] \leq \epsilon$ が成り立つとき, アルゴリズム $(Train, Infer)$ は ϵ-accurate と呼ぶ. 直観的には ϵ が低いほど, 良いアルゴリズムといえる.

2.2 機械学習へのバックドア

機械学習モデルにおけるバックドアは, 入力 x において誤ったラベル l_w を推論するようモデルを操作する手法である [9]. このような誤った推論をさせる入力集合 $T \subset D$ をトリガ集合と呼ぶ. また, この誤った推論をする関数をトリガ関数と呼び, $f_w : T \to L \backslash \{\perp\}; x \in T \mapsto f_w(x) \neq f(x)$ と定義する. このとき, 上述したトリガ集合 T とトリガ関数 f_w を合わせてバックドア $b = (T, f_w)$ と呼ぶ.

このとき, バックドアはアルゴリズム $SamBackdoor$ と $Backdoor$ から定義される. $SamBackdoor$ は入力として目的正解関数 f を与えられ, バックドア b を出力する. 次に, $Backdoor$ は f, モデル M, バックドア b を与えられ, トリガー集合上で誤分類するモデル M' を出力する. M' が $\hat{D} \backslash T$ において $\Pr_{x \in \hat{D} \backslash T}[f(x) \neq Infer(M', x)] \leq \epsilon$, かつ, $\Pr_{x \in T}[f_w(x) \neq Infer(M', x)] \leq \epsilon$ を満たすなら, M' は ϵ-バックドア化されるという.

3. 機械学習への電子透かし

本節では外部機構を伴う電子透かしの要件を定義する. 既存研究 [15] でも要件は示されているが, 本稿ではオラクルを用いた計算論的定式化により, 外部機構の要件をより厳密に定義する. 以降ではまず前提条件を述べる. 次に, その条件の下で, 電子透かしの安全性を定式化する.

3.1 前提条件

機械学習への電子透かしでは, モデルの挙動に電子透かしを埋め込む [1], [17]. 具体的には, 透かしに関する鍵情報をモデルに埋め込むことで, 透かしの検証用鍵情報を推論処理として入力した際に, あらかじめ想定された出力を返すよう設定する. このとき, 一般には, 透かし埋め込み用の鍵として機械学習モデルに付随するパラメータとモデルへの入力（すなわち, 学習データおよび推論時の入力）を用いる. また, 画像など従来のコンテンツ保護と比べて, 透

かしの埋め込みによりモデルの推論精度が劣化しないこと, および, モデルは埋め込み後にも継続して学習などが行われることから, モデル内のパラメータが変化した後でも透かしが残っていることが求められる [13].

この設定において, 参加者はサービス提供者とユーザの二つである. サービス提供者は推論処理を行う機械学習モデルを提供しており, ユーザはそのモデルの API を介してモデルへ入力を与え, モデルからの出力結果を受け取る. ユーザによるモデルの二次利用が行われていた場合, サービス提供者は透かしの検証用鍵情報を入力することで, その出力を通じて自らのモデルか確認する.

上述した前提に加え, 本稿で着目する外部機構を用いる電子透かし [15] では, モデルと透かしそのものを個別に設ける. つまり, 内部機構を用いた方式 [1], [17] が学習を通じてモデルの内部パラメータに透かしを埋め込むことに対し, 外部機構では内部パラメータを操作することは考えず, モデルに対しフロントエンドで動作する. このとき, 仮定として, 外部機構ではユーザの手元にモデルを直接配布することは考えない. これは例えばユーザがリバースエンジニアリングなどで外部機構を取りはずすことで, 透かしの機構を回避できるためである. このユースケースとしては, サービス提供者がサーバ上にある機械学習モデルのインターフェースの利用権利をユーザに販売することで, ユーザが任意に利用できる状況が挙げられる.

3.2 電子透かしの要件定義

電子透かしは一般に三つのアルゴリズム $(KeyGen, Mark, Ver)$ から定義される. $KeyGen$ は鍵生成を行うアルゴリズムであり, セキュリティパラメータ 1^k を与えられ, 埋め込み鍵 mk と検証鍵 vk を出力する. $Mark$ は透かしを埋め込むアルゴリズムであり, 透かしを埋め込む入力コンテンツ X と埋め込み鍵 mk を与えられ, 透かしを埋め込んだコンテンツ $\bar{X}$ を出力する. Ver は埋め込み鍵 mk, 検証鍵 vk, コンテンツ X を与えられ, $\top$ か $\bot$ を出力する.

このとき, 電子透かしの正当性は $(mk, vk) \leftarrow KeyGen(1^k)$ および任意の入力 X において, $\Pr[Ver(mk, vk, Mark(X, mk)) = \top] = 1$ が成り立つこととして定義される. 機械学習においては, X としてモデル M を入力する. このとき, さらに, $\Pr_{x \in \hat{D}}[Infer(x, M) = f(x)] \approx \Pr_{x \in \hat{D}}[Infer(x, \bar{M}) = f(x)]$ となることが正当性の要件として定義される.

3.3 安全性の定義

本稿では電子透かしが埋め込まれたモデルに対し, モデル提供者があらかじめ設定した推論結果と異なる結果を返すように攻撃者が操作することで, 電子透かしを取り除く状況を考える. ここで透かしの除去とは, 埋め込み鍵 m と検証鍵 vk をモデルに入力しても, 設定されたような推論結果にならないことを意味する. これによりモデル提供者による所有権の確認を回避することができるようになる.

このような脅威シナリオとして, 推論処理の出力から攻撃者が自ら透かしを埋め込んだモデルを手元で作成する**偽造不可能性**, および, 攻撃者が選んだデータによる学習を介してサービス提供者のホストするモデルから透かしを除去する**除去不可能性**をそれぞれ以下に定式化する.

まず攻撃者が利用可能なオラクルとして, 学習オラクル $\mathcal{O}_T$ と推論オラクル $\mathcal{O}_I$ を定義する. $\mathcal{O}_T$ はあらかじめ定められた目的関数 f とモデル M に関して, 攻撃者が学習データ $\boldsymbol{X} = \{x_1, ..., x_n\}$ とそのラベル $\boldsymbol{L} = \{l_1, ..., l_n\}$ を入力することで, 新たなモデル M' を返す. 次に, $\mathcal{O}_I$ はあらかじめ定められたモデル M に従い, 攻撃者が推論対象のデータ x を入力することで, 推論結果 $M(x)$ を返す. これらのオラクルのもとで, 各安全性はチャレンジャ $\mathcal{C}$ と確率的多項式時間攻撃者 $\mathcal{A}$ のゲームとして定式化される.

3.3.1 偽造不可能性

偽造不可能性は以下のとおり定義される.

初期設定 $\mathcal{C}$ は $(mk, vk) \leftarrow KeyGen(1^k)$ を生成する. 次に, モデル M, および透かしを埋め込んだモデル $\bar{M} \leftarrow Mark(M, mk)$ を作成する. $\bar{M}$ に関する推論オラクル $\mathcal{O}_I$ へのアクセスが認められる $\mathcal{A}$ に対し, $\mathcal{C}$ は vk を入力する.

クエリ $\mathcal{A}$ は $\mathcal{O}_I$ へのクエリを多項式回行える.

出力 $\mathcal{A}$ は自らの埋め込み鍵 mk^* とモデル M^* を出力する. このとき, $\mathcal{C}$ は $Ver(mk^*, vk, M^*) = \top$ が成り立つか確認する. もし成り立つなら $\mathcal{A}$ の勝利となる.

Definition 1. $\mathcal{A}$ が上述したゲームにおいて確率 ϵ 以上で勝利するなら, $\mathcal{A}$ は偽造不可能性を ϵ で破るという. そのような任意の確率的多項式時間攻撃者が存在しないなら, 電子透かしは ϵ-偽造不可能性を満たす.

3.3.2 除去不可能性

除去不可能性は以下のとおり定義される.

初期設定 $\mathcal{C}$ は偽造不可能性と同様に $(mk, vk, M, \bar{M}$ を作成する. $\bar{M}$ に関する学習オラクル $\mathcal{O}_T$ へのアクセスが認められる $\mathcal{A}$ に対し, $\mathcal{C}$ は vk を入力する.

クエリ $\mathcal{A}$ は $\mathcal{O}_T$ へのクエリを多項式回行える.

出力 $\mathcal{A}$ はモデル M^* を出力する. このとき, $\mathcal{C}$ は $Ver(mk^*, vk, M^*) = \bot$, かつ, $\Pr_{x \in D}[Infer(x, M) = f(x)] \approx \Pr_{x \in D}[Infer(x, M^*) = f(x)]$ が成り立つか確認する. もし成り立つなら $\mathcal{A}$ の勝利となる.

Definition 2. $\mathcal{A}$ が上述したゲームにおいて確率 ϵ 以上で勝利するなら, $\mathcal{A}$ は除去不可能性を ϵ で破るという. そのような任意の確率的多項式時間攻撃者が存在しないなら, 電子透かしは ϵ-除去不可能性を満たす.

3.3.3 実験的評価対象の安全性

以降では, 以下にモデル抽出攻撃 [16] および再学習への

Vol.2020-CSEC-91 No.22
Vol.2020-SPT-40 No.22
Vol.2020-EIP-90 No.22
2020/11/26

耐性 [17] を示す. 前節で述べた安全性は透かしの検証が成り立つことを前提とした安全性だった. これらに対し, モデル抽出攻撃および再学習は, 透かしの検証を前提としないような, 偽造不可能性および除去不可能性に関する特殊な場合といえる. すなわち, 計算論モデルにおける議論は難しい一方, モデルの実用上において重要な安全性といえる. 以下にそれぞれの詳細について述べる.

3.3.3.1 モデル抽出攻撃

モデル抽出攻撃 [16] は, サービスとして提供されているモデルの入出力を通じて, 攻撃者が自らの手元に同等程度の性能を持つモデルを複製する攻撃である. 安全性は, 最後の勝利条件を除いて偽造不可能性と同様に定義される [4]. 具体的には, $\Pr_{x\in D}[Infer(x,M) = f(x)] \approx \Pr_{x\in D}[Infer(x,M^*) = f(x)]$ を攻撃者の勝利条件とする. この定義は, 透かしの検証結果に依存せず, ただ精度の良いモデルを攻撃者が出力することを意味する. このため, 攻撃者がモデル M の学習データ $\boldsymbol{X}$ に対し十分なクエリを用いることで, 満たすことができる. 本項ではモデル抽出攻撃については実験的に評価する.

3.3.3.2 再学習への耐性

再学習への耐性とは, サービスとして提供されているモデルに埋め込まれた電子透かしが, モデルの汎化性能を改善させる再学習に対して頑健であることを示す [17]. この再学習を攻撃者が行うことで, 学習を通じて透かしを除去することも起こりえる. 再学習への耐性は, 最後の勝利条件を除いて除去不可能性と同様に定義される. 具体的には, $Ver(mk^*, vk, M^*) = \perp$ を攻撃者の勝利条件とする. この定義は, 攻撃者が得るモデルの性能を考慮することなく, 透かしの検証を回避することを意味する. 再学習への耐性についても, 実験的に評価する.

3.4 技術的課題

外部機構による構成は, 既存の内部機構による方式 [1], [17], [21] の知見が応用できず, 非自明である. 直観的には, 外部機構ではモデルの内部パラメータを触ることができず, モデルをブラックボックスとして扱った状態で透かしを埋め込む必要がある. すなわち, サービス提供者は透かしの埋め込みとしてできる操作が出力操作に限られる状況において, 攻撃者が透かしのないモデルを得られないようにすることが求められる.

このとき, 攻撃者が出力するであろうモデルに透かしを埋め込むためには, 透かしのパラメータを明確に定める必要がある. パラメータが定まっていない限り, 直観的にはモデルの入出力の分布はランダムにならざるを得ない. このため, 既存研究では入力に対する HMAC を計算することで, その出力を一様ランダムにしていた. しかし, このような手法では攻撃者のモデルに対し, 透かしを埋め込めるとは限らない. すなわち, 偽造不可能性, 特にモデル抽出攻撃への耐性が損なわれてしまう[*2].

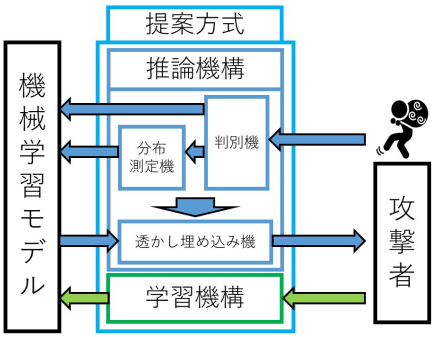


図 1: 本稿における構成の全体像

4. 提案方式

本節では本稿で提案する外部機構に基づく電子透かしの方式を説明する. 本節ではまず提案方式の全体像を述べたのち, 各アルゴリズムについて述べる.

4.1 提案方式の全体像

機械学習の処理は学習と推論からなるため, 提案方式においても学習用クエリを受けてモデルを学習させる “学習機構” と, 推論用クエリを受けてモデルが結果を返す “推論機構” の二つの機構を持つ. この全体像を図 1 に示す.

まず学習機構の操作として, サービス利用者からのデータ提供を受けた際に, モデル自体が持つ学習アルゴリズムを用いることで, モデルを再学習する. これにより, モデルの精度向上を継続的に支援する.

次に, 推論機構は判別機, 分布判別機, 透かし埋め込み機の三つからなる. 推論機構の操作として, サービス利用者から推論クエリを受けた際に, まず判別機が 2.2 節で述べたバックドアとして設定されたトリガ集合に存在する入力か判別する. トリガ集合に存在するデータと一致している場合, すなわち判別結果が真の場合, モデルにクエリを転送する. 一致していない場合, 分布測定器がトリガ集合に分布が近いデータか判別する. 分布測定器は判別結果に問わず, モデルにクエリを転送する. このとき, 分布判別機はモデルへの転送に並行して, 推論クエリとして与えられた入力分布とトリガ集合に属するデータの分布の差が, あらかじめ定められたしきい値内に収まるか判別する. 判別機あるいは分布測定器いずれかの判別結果が真の場合, モデルから結果が返ってきた際, その結果をトリガ集合に対する結果に一致するように出力を操作してからサービス利用者に返す. これにより, サービスを提供するとともに, サービス利用者にトリガ集合の情報, すなわち電子透かしを持ち帰らせるようになる.

以降ではモデル M の出力は任意の整数 $i \in \mathbb{N}$ に関して

[*2] DAWN [15] では偽造不可能性は議論されているが, 扱っている攻撃の性質上, 偏ったデータ分布をクエリしている可能性がある.

$\{l_1, \cdots, l_k\}$ としている. そこでの l_i はその確信度を意味し, それぞれの合計が 100%となるようになっている. また, トリガ関数 f_w の出力は $i \in \mathbb{N}$ をとる.

4.2 初期設定：KeyGen

初期設定として $KeyGen$ の処理を以下に記載する. まず $SamBackdoor$ を用いることで, 目的正解関数 f に関するバックドア $b = (T, f_w)$ を生成する. ここで T は任意個の要素 $\{t_1, \cdots, t_n\}$ からなるトリガ集合, f_w を T に関して f と異なった推論をするトリガ関数とする. また, セキュリティパラメータ k において, 暗号学的鍵付きハッシュ関数 $H_K : \{0,1\}^* \to \{0,1\}^k$ を選ぶ. このとき, 埋め込み鍵 $mk = (b, H_K)$, 透かし検証鍵 $vk = \{T, \{f_w(t)\}_{t \in T}, \{H_K(t)\}_{t \in T}\}$ となる.

4.3 透かし埋め込み：Mark

透かし埋め込み処理 $Mark$ では, 以下に示す判別機, 分布測定器, 透かし埋め込み機をモジュールとして用いる. それぞれの詳細を以下に述べる. 以降ではトリガ集合 T に属するデータを任意の $i \in \mathbb{N}$ を用いて $t_i \in T$ とする. ここで, $\mathbb{N}$ は整数集合とする.

4.3.1 判別機

判別機は与えられた推論クエリ x が埋め込み鍵 mk に存在するデータか判別する. このとき, $\exists t_i \in T, H_K(x) = H_K(t_i)$ となるような x が埋め込み鍵に存在するか確認する. 成り立つ場合, 与えられたクエリ x は透かし埋め込み対象のクエリとみなす. その場合, 後述する透かし埋め込み機に t_i を渡し呼び出すとともに, x をモデル M に渡す. また, $H(x) \neq H(t_i)$ ならば, x を分布測定機に渡す.

4.3.2 分布測定機

分布測定機は判別機で透かし埋め込みではないと判断された推論クエリ x に対し, その分布が埋め込み鍵 mk に存在するデータの分布に似ているか判別する. 似ている場合, x は透かし埋め込み対象とみなす. 具体的には, 任意のデータに対する平均を E, 分散を V としたとき, $\exists t_i \in T, (|E[t_i] - E[x]| < \delta) \vee (V[t_i] - V[x]| < \delta)$ となるような $t_i \in T$ が存在するか確認する. ここで, δ はモデル提供者があらかじめ設定できる任意のパラメータである. 存在するならば, 透かし埋め込み機を x 渡して呼び出すとともに, x をモデル M に渡す. 存在しない場合, x をモデル M に渡すのみである.

4.3.3 透かし埋め込み機

透かし埋め込み機は判別機あるいは分布測定機による呼び出しに従い, トリガ集合に属するデータ, あるいは近い分布を持つデータが入力された際に, モデル M からの推論結果を操作することでモデル M に対して渡された数値を電子透かしとして埋め込む.

このとき, 透かし埋め込み機は, 推論結果において確率が最大になっているラベル項目 $l_{max} \in \{l_1, \cdots, l_k\}$ において, 以下の処理を行う. まず l_{max} に対応する整数を $i' \in [1, k]$ とする. 次に, 初期設定で述べたトリガ関数 f_w の出力から, x に相当する $x \in T$ の出力 $f_w(x)$ を選ぶ. その後に, $f_w(x)$ として算出される l'_i と $l_{f_w(x)}$ の数値を入れ替える. これにより, $i' = f_w(x)$ となり, それに相当する $l_{i'}$ が最大値となるように出力が設定される. その後, 残りの項目 $\{l_1, \cdots, l_{k-1}\} \backslash \{l_{max}\}$ を昇順にソートする. そのソート済みの結果を, 透かしを埋め込んだ応答 $M'(x)$ として返す.

4.4 透かしの検証：Ver

検証処理では, 検証対象のモデル M に電子透かしが存在するか確認する. このとき, まず mk に属する $t'_i \in T$ および vk に属する $H_K(t_i)$ において, まず $H_K(t'_i) = H_K(t_i)$ が成り立つか確認する. 成り立たない場合, 処理を中断する. 成り立つ場合, $M_v(t_i)$ に関して最大の値をとるラベルに対応する値は $i' = f_w(t_i) \in vk$ とする. これに対し, M でも同様に $M(t_i)$ の最大の値をとるラベルに対応する値が $i' = f_w(t_i) \in vk$ となるとき, M において t_i に関する透かしを確認したとみなす. 確認できた透かしの数を m, バックドアのトリガ集合 T の要素数を $|T|$ としたとき, $\frac{m}{|T|} > \gamma$ が成り立つか確認する. 成り立つならば, 電子透かしの検証がされたとし, $\top$ を出力する.

安全性の証明：以下に提案方式が正当性, 偽造不可能性, 除去不可能性を満たすことを述べる. 紙面上詳細は割愛するが, 以下の議論は文献 [1] 同様に示すことが可能である[*3]. まず正当性は 2.2 節で述べた ϵ-バックドアを用いることで保証される. その具体的な手法としては文献 [9] を用いることが可能である. 次に, 偽造不可能性は鍵付きハッシュ関数の衝突困難性に帰着される. これは検証において mk に属する $t \in T$ と vk に属するそのハッシュ値 $H_K(t)$ の確認をはさむこと, また, これらの値は H_K が攻撃者に未知である限り, 計算できないことが直観である. 同様に, 除去不可能性は鍵付きハッシュ関数がランダムオラクル, つまり出力が一様ランダムと仮定したとき, mk に属する $t \in T$ と vk に属するハッシュ値を $t' \neq t$ なるハッシュ値 $H_K(t')$ に一つずつ置き換えることにより, 攻撃者のアドバンテージを下げることで示される.

5. 実験

本節では提案方式の実験評価について述べる.

5.1 実験目的

提案方式の性能評価として, モデル抽出攻撃および再学習をモデルに対して実施することで, 提案方式におけるこれらの攻撃に対する耐性を評価する. 具体的には, 透かし

[*3] 文献 [1] ではコミットメント方式を用いているが, コミットメントを鍵付きハッシュ関数に置き換えることで同様に証明できる.

を持たない一般的なモデル及び内部機構の方式 [1] と併せて実験する．これにより, 外部機構による攻撃耐性及び内部機構と比べた特徴を確認する．

まずモデル抽出攻撃を行うことで, 攻撃者によるモデルへのクエリを通じて, 攻撃者がどの程度の精度のモデルを作れるか, また, モデル抽出によって透かしが攻撃者のモデルに残せるか確認する．このとき, 攻撃者が複製したモデルからあらかじめ設定した透かしが確認できれば, モデル抽出攻撃への耐性があるといえる．

次に再学習を行うことで, モデルの推論に応じてモデルの更新が生じた際に, 透かしがモデルから取り除かれるか確認する．とくに, 内部機構の方式と攻撃後にモデルに残留する透かしの割合を比較することで, 再学習に対する外部機構の有効性を確認する．また, 提案方式のパラメータ δ を変化させて計測することで, 再学習における精度への影響についても評価する．

5.2 実験設定

本実験は一般的な性能評価として画像分類を対象に行う．

アーキテクチャとデータセット：アーキテクチャに ResNet18, データセット CIFAR-10[*4] ベンチマークを利用した三つのモデルを作成する．一つ目は単に CIFAR-10 を学習させた電子透かしのない一般的なモデル (以下, 透かし無しモデルと呼ぶ), 二つ目は内部機構として作成した既存研究として WatermarkNN [1] のモデル (以下, 内部機構と呼ぶ) である．三つ目は 4 節の提案方式を導入したモデルである．

一種類目及び二種類目のモデルをベースラインとする．このとき, 提案方式のパラメータである *delta* を変化させながら, モデル抽出攻撃と再学習への耐性について, それぞれのモデルの性能を比較する．

なお, トリガ集合として, 学習用データセットとは別に選出した画像を用いる．これらの画像な中に「TEST」という文字画像を埋め込み T とし, 対応する $f_w(T)$ を「トラック」とラベル付けをしてデータセットとする．既存文献 [21] によると, この方法が電子透かしとして汎用的な性能を提供できる．このようなデータセットを 100 件用意し, 提案方式の内部に設けた．

モデル抽出攻撃の設定：モデル抽出攻撃の設定として, 攻撃者はサービス提供者が学習に利用したデータセット 50000 件の内, 10000 件を持っているものとする．攻撃者が自ら学習させるモデルには ResNet18 を使用した．このとき, サービス提供者が持つ前述した三つのモデルそれぞれへのクエリ及びその応答結果に対して, 50 エポックずつ学習させる．この攻撃者のモデルの精度を計測する．直観として, 最終的に示されるテスト集合の数値が低いほど, また, トリガ集合の数値が高いほど攻撃耐性があるといえる．

これにより, 三つのモデルに対してモデル抽出攻撃がどのように働くか確認する．

再学習への耐性の設定：再学習の設定として, 攻撃者はモデルに用意された学習機構として, 最終層再学習 (FTLL) と全層再学習 (FTAL) を使用する．これらの設定は WatermarkNN [1] の実験設定に準拠している．再学習方法は最終層以外のモデルのパラメータを固定し, 学習により最終層パラメータを更新する**最終層再学習** (FTLL) と, 学習によりすべてのパラメータを更新する**全層再学習** (FTAL) を利用する．これらアルゴリズムは一般にモデルの汎化性能を上げるものである．このため, 学習時に使用した CIFAR-10 データセットを利用し, 20 エポック学習させた．

このとき, 各モデルにおいてテスト集合およびトリガ集合それぞれの精度を計測する．直観として, 最終的に示されるテスト集合の数値が高いほど, また, トリガ集合の数値が低いほど再学習耐性があるといえる．厳密には, 汎化性能を向上させる再学習では精度自体は下がることが予想される．このとき, 下がり幅が小さいほど耐性がある．

5.3 結果

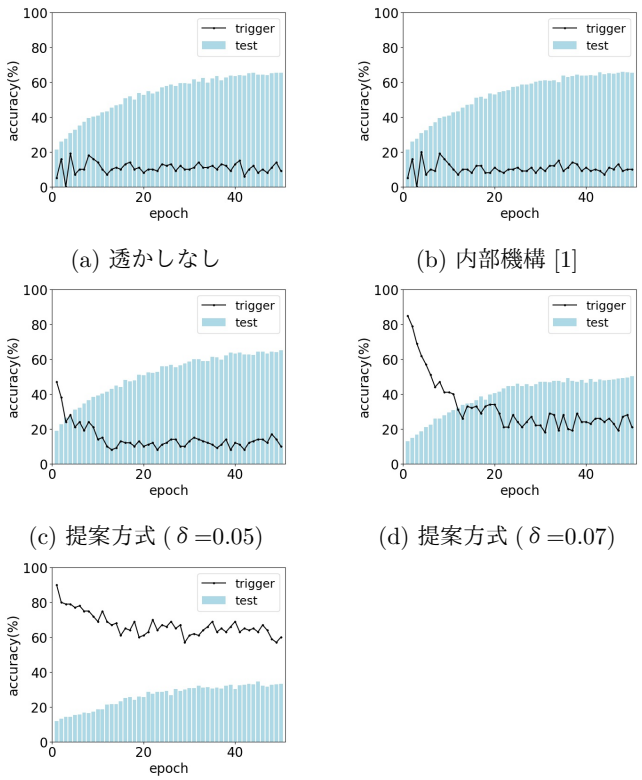
モデル抽出攻撃の結果：透かしなしモデル, 内部機構, 及び提案方式それぞれに関するモデル抽出攻撃の結果を図 2 に示す．各図において, 折れ線は攻撃者がクエリから学習させるモデルに関して, トリガ集合に対するエポック毎の推論精度を表す．また, 棒グラフは CIFAR-10 内のテスト集合に対する推論精度を示している．

まず, 図 2b, 図 2c によると, 各モデルの結果は図 2a と同様の数値を示していることから, 内部機構と提案方式 ($\delta = 0.05$) はモデル抽出攻撃に対して耐性がないといえる．それに対して, 図 2d ではテスト集合の精度が 15% ほど下がっており, また, トリガ集合の精度が 10% ほど上がっている．さらに, 図 2e では, テスト集合の精度は 15%下がり, 35% にまでなっている一方, トリガ集合の精度は 60%を示している．これらの事から提案方式では, δ を大きく設定することで, 攻撃者のモデルにトリガ集合を埋め込むこと, すなわちモデル抽出攻撃に耐性があることがわかった．

再学習への耐性の結果：各モデルにおける再学習の結果を表 1 に, 提案方式におけるパラメータ δ 毎の精度の影響を表 2 にそれぞれ示す．

表 1 によると, 全体的に再学習によってテスト集合の精度は下がっているが, 前述した通り, これらは汎化性能を上げた影響である．このため, 精度の下がり幅が性能指標となる．まず, 内部機構, 内部機構 FTLL, 内部機構 FTAL を比較すると, 内部機構として埋め込まれたトリガ集合は, FTAL でのみ除去されている．これは, 内部機構のモデルはパラメータを大きく操作するような運用は難しいことを意味している．一方, 提案方式では FTLL と FTAL いずれ

[*4] https://www.cs.toronto.edu/~kriz/cifar.html



(a) 透かしなし　(b) 内部機構 [1]

(c) 提案方式 (δ =0.05)　(d) 提案方式 (δ =0.07)

(e) 提案方式 (δ =0.09)

図 2: モデル抽出攻撃に関する結果

表 1: 再学習への耐性に関する結果

モデル	テスト集合 [%]	トリガ集合 [%]
透かしなし	91.990	4.000
透かしなし FTLL	91.940	4.000
透かしなし FTAL	88.940	4.000
内部機構 [1]	91.660	100.00
内部機構 [1] FTLL	91.570	100.000
内部機構 [1] FTAL	88.070	5.000
提案方式 (δ=0.05)	85.680	100.000
提案方式 (δ=0.05) FTLL	85.620	100.000
提案方式 (δ=0.05) FTAL	82.800	100.000

表 2: 各 δ とテスト集合の精度

パラメータ δ	0.01	0.03	0.05	0.07	0.09
テスト集合 [%]	91.740	89.540	85.680	80.510	74.100

においても高い耐性を持つことが示されている．これは外部機構の直観と一致する．また, 透かしなしモデルと比較してテスト集合における精度の低下がみられるが, これは表 2 によると分布測定機による出力操作の影響である．つまり, δ の値を通じて精度を柔軟に制御できている．

6. 考察

6.1 実験結果に関して

前節で示した結果によると, 外部機構は電子透かしを埋め込む機能がモデルと独立していることから, 内部機構に比べ再学習への耐性に優れている．これは推論機構が呼び出されるたびに確実に電子透かしの埋め込みができるためである．また, 提案方式は δ の値を大きくすることで, モデル抽出攻撃に対しても高い耐性を示している．なお, モデル抽出攻撃において攻撃者の精度が低下している点には, モデルの出力結果をソートしていることが要因として考えられる．CIFAR-10 の特徴量を鑑みると, 今回の結果では $\delta = 0.07$ のときがモデル抽出攻撃および再学習への耐性双方を満たし, かつ, テスト集合の精度も比較的維持できていることから, 最適なパラメータといえる．

6.2 制約条件

提案方式における現状の制約条件を以下に示す．まずは精度への影響である．表 1, 表 2 に示した通り, 内部機構と比べて精度に若干の低下がみられる．これはモデルからの出力を直接制御していることに起因するため, 外部機構の本質的な問題ともいえる．

次に, 攻撃者が学習機構を通じてバックドアを新たに設けられる可能性がある．提案方式では学習機構のインターフェースも提供していることから, 攻撃者は自ら透かしとなるような (mk, vk) をバックドア攻撃 [8] を用いて埋め込むことができる．これは攻撃者自ら権利を主張するような所有権侵害 [1], [11] にもつながるため, 対策が必要となる．

7. 関連研究

機械学習モデルへの電子透かしにおける代表的な枠組みとして, モデルパラメータへ直接埋め込む手法と, トリガ集合を用いた手法をそれぞれ説明する．更なる興味を持つ読者は文献 [3] を参照されたい．

モデルパラメータへの埋め込み：最初のモデルへの電子透かし [17] は学習の損失関数と埋め込み時にパラメータを正則化する関数を合成することで, 透かしを埋め込んだ．この応用として, 透かし用にサブモデルの利用 [18], [19] や, 隠れ層を設ける方法 [7] も示されている．しかし, パラメータへ直接埋め込む方式はモデルのアーキテクチャごとに設計されており, アーキテクチャが変わった際はその都度手法を設計する必要がある．これはアーキテクチャごとの互換性という観点で, トリガに基づく方式, とくに外部機構を用いる構成の方が優れていることを意味する．

トリガに基づく方式：近年の研究では, トリガ集合をモデルに学習させる手法 [1] を採用している．この応用として, 敵対的サンプルで学習する方法 [12] が知られているが, これは透かしのないモデルを透かし有りと誤認識する問題がある [14]．モデルの内部構造をブラックボックスとして透かしを埋め込む方法 [5] も知られている．また, 近年ではトリガを用いた構成の拡張として, 短いビット列からなる透かしの組み込み機器上での再学習 [10] や, 連合学習の中で透かしを埋め込む手法 [2] も提案されている．これらは

いずれも内部機構に基づく方式である.

外部機構を用いた構成としては, 著者らが知る限り DAWN [15] が唯一の既存方式である. しかしながら, 1.2 節で述べたとおり, DAWN はパラメータ設計を議論しておらず, また, 本稿で議論しているような内部機構との比較は議論されていない.

なお, データ分布の平均値を学習させることで確率密度関数に埋め込む方式 [6] もあり, これは攻撃者に気づかれにくいことも示されている. 本稿の分布測定器にも, この結果は利用されている. また, 本稿で用いたトリガ画像の生成手法 [21] の改良版として, 元画像を崩さずにロゴを埋め込む方法 [11] も知られている. この方法を応用することで, 本稿で示した実験精度の改善が期待できる.

8. まとめ

本稿では外部機構による機械学習モデルへの電子透かしにおいて, 透かしの埋め込みに関するパラメータ設計を考慮した方式を提案した. 既存の外部機構方式 [15] では, パラメータの設計は議論されておらず, 実装や安全性の解析が十分に行えなかった. 対して, 本稿ではパラメータの設計を通じて, その具体的構成を示すとともに, 実装を通じてモデル抽出攻撃 [16] や再学習への耐性 [17] を実験的に確認した. 今後の課題は, 内部機構 [1] と比較した精度の向上にある. とくにトリガ相当のクエリをより正確に判別すべく, 分布測定機に Jaccard 係数の利用を検討している.

謝辞 本研究開発の一部は, 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）にて実施されております.

参考文献

[1] Adi, Y., Baum, C., Cisse, M., Pinkas, B. and Keshet, J.: Turning Your Weakness Into a Strength: Watermarking Deep Neural Networks by Backdooring, *Proc. of USENIX Security 2018*, USENIX Association, pp. 1615–1631 (2018).

[2] Atli, B. G., Xia, Y., Marchal, S. and Asokan, N.: WAFFLE: Watermarking in Federated Learning, *CoRR*, Vol. abs/2008.07298 (online), available from ⟨https://arxiv.org/abs/2008.07298⟩ (2020).

[3] Boenisch, F.: A Survey on Model Watermarking Neural Networks, *CoRR*, Vol. abs/2009.12153 (online), available from ⟨https://arxiv.org/abs/2009.12153⟩ (2020).

[4] Carlini, N., Jagielski, M. and Mironov, I.: Cryptanalytic Extraction of Neural Network Models, *Proc. of CRYPTO 2020*, LNCS, Vol. 12172, Springer, pp. 189–218 (2020).

[5] Chen, H., Rouhani, B. D. and Koushanfar, F.: BlackMarks: Blackbox Multibit Watermarking for Deep Neural Networks, *CoRR*, Vol. abs/1904.00344 (online), available from ⟨http://arxiv.org/abs/1904.00344⟩ (2019).

[6] Darvish Rouhani, B., Chen, H. and Koushanfar, F.: DeepSigns: An End-to-End Watermarking Framework for Ownership Protection of Deep Neural Networks, *Proc. of ASPLOS 2019*, ACM, p. 485–497 (2019).

[7] Fan, L., Ng, K. W. and Chan, C. S.: Rethinking Deep Neural Network Ownership Verification: Embedding Passports to Defeat Ambiguity Attacks, *Proc. of NIPS 32*, Curran Associates, Inc., pp. 4714–4723 (2019).

[8] Gao, Y., Doan, B. G., Zhang, Z., Ma, S., Zhang, J., Fu, A., Nepal, S. and Kim, H.: Backdoor Attacks and Countermeasures on Deep Learning: A Comprehensive Review, *CoRR*, Vol. abs/2007.10760 (online), available from ⟨https://arxiv.org/abs/2007.10760⟩ (2020).

[9] Gu, T., Dolan-Gavitt, B. and Garg, S.: BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain, *CoRR*, Vol. abs/1708.06733 (online), available from ⟨http://arxiv.org/abs/1708.06733⟩ (2017).

[10] Guo, J. and Potkonjak, M.: Watermarking Deep Neural Networks for Embedded Systems, *Proc. of ICCAD 2018*, ACM, pp. 1–8 (2018).

[11] Li, Z., Hu, C., Zhang, Y. and Guo, S.: How to Prove Your Model Belongs to You: A Blind-Watermark Based Framework to Protect Intellectual Property of DNN, *Proc. of ACSAC 2019*, ACM, p. 126–137 (2019).

[12] Merrer, E. L., Pérez, P. and Trédan, G.: Adversarial frontier stitching for remote neural network watermarking, *Neural Computing and Applications*, Vol. 32, No. 13, pp. 9233–9244 (2020).

[13] Nagai, Y., Uchida, Y., Sakazawa, S. and Satoh, S.: Digital watermarking for deep neural networks, *International Journal of Multimedia Information Retrieval*, Vol. 7, No. 1, pp. 3–16 (2018).

[14] Namba, R. and Sakuma, J.: Robust Watermarking of Neural Network with Exponential Weighting, *Proc. of AsiaCCS 2019*, ACM, pp. 228–240 (2019).

[15] Szyller, S., Atli, B. G., Marchal, S. and Asokan, N.: DAWN: Dynamic Adversarial Watermarking of Neural Networks, *CoRR*, Vol. abs/1906.00830 (online), available from ⟨http://arxiv.org/abs/1906.00830⟩ (2019).

[16] Tramér, F., Zhang, F. and Juels, A.: Stealing Machine Learning Models via Prediction APIs, *Proc. of USENIX Security 2016*, USENIX Association, pp. 601–618 (2016).

[17] Uchida, Y., Nagai, Y., Sakazawa, S. and Satoh, S.: Embedding Watermarks into Deep Neural Networks, *Proc. of ICMR 2017*, ACM, p. 269–277 (2017).

[18] Wang, T. and Kerschbaum, F.: Attacks on Digital Watermarks for Deep Neural Networks, *Proc. of ICASSP 2019*, IEEE, pp. 2622–2626 (2019).

[19] Wang, T. and Kerschbaum, F.: Robust and Undetectable White-Box Watermarks for Deep Neural Networks, *CoRR*, Vol. abs/1910.14268 (online), available from ⟨http://arxiv.org/abs/1910.14268⟩ (2019).

[20] Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R. and Le, Q. V.: XLNet: Generalized Autoregressive Pretraining for Language Understanding, *Proc. of NeurIPS 2019*, Curran Associates, Inc., pp. 5753–5763 (2019).

[21] Zhang, J., Gu, Z., Jang, J., Wu, H., Stoecklin, M. P., Huang, H. and Molloy, I.: Protecting Intellectual Property of Deep Neural Networks with Watermarking, *Proc. of AsiaCCS 2018*, ACM, p. 159–172 (2018).