マルウェア検知のためのpAUC最大化学習法

西山 泰史[1]　熊谷 充敏[1]　藤野 昭典[2]　神谷 和憲[1]

概要：マルウェアの感染被害を抑制するため，機械学習を用いて通信ログを自動で分析し，マルウェア由来の通信を検知する手法が注目されている．通常，通信ログは大量の正常ログと少量のマルウェア由来のログからなる不均衡なデータであるため，多くの先行研究では検知性能を測る指標として，不均衡なデータに適した指標である Area Under the Curve（AUC）を用いている．しかし，実運用ではネットワーク管理者の負担軽減の観点から，AUC 全体ではなく，低誤検知領域の AUC が重要となる．機械学習分野では，任意の一部の AUC を直接最大化する手法がいくつか提案されているが，これらの研究では，分類器によって付与された学習データのスコアに引き分けがないことを前提にしていた．一方，通信ログ分析においては，ロードバランサ等の影響で，特徴ベクトル化した際に引き分けが生じうる事例が多数存在する．そこで本稿では，スコアの引き分けを考慮して，任意の誤検知領域の AUC を最大化する学習法を提案する．これにより，従来法より正確に任意の誤検知領域の AUC を最大化することができる．また，実在の大企業網の proxy ログを用いて，既存手法と性能比較を行い，提案法の効果を示す．

pAUC Maximization Method for Malware Detection

Taishi Nishiyama[1]　Atsutoshi Kumagai[1]　Akinori Fujino[2]　Kazunori Kamiya[1]

Abstract: Machine learning is becoming a vital component to automatically analyze network logs and detect malicious activities for mitigating damage of malware infection. Since actual network logs are imbalanced data that contain a small amount of malicious logs compared to benign logs, many studies have evaluated the classification performance by using the area under the curve (AUC). However, in actual security operation, a low false positive rate is required to reduce the burden on network operators. Therefore, the area to be focused on is not the entire AUC but the partial AUC in a specific low false potive rate (FPR) interval. In the field of machine learning, several machine learning studies described the methods for directly maximizing an arbitrary area of AUC. However, they assumed that there is no tie in the score of training data, which is not always appropriate in network log analysis since there are many ties, e.g., when the network logs with a load balancer are converted into feature vectors. In this paper, we propose a novel method for maximizing partial AUC in an arbitrary FPR interval with considering ties, which maximizes partial AUC in an arbitrary FPR interval more accurately than conventional methods. We also show the effectiveness of our method after comparing it with conventional methods with proxy logs from a real-world enterprise network.

1. はじめに

マルウェアの感染被害を抑制するためのセキュリティ対策として，ネットワーク内の通信ログを分析して感染を早期に検知する手法が活用されている [1]．近年，検知ロジックの高度化やネットワーク管理者の負担軽減を目的として，機械学習を用いた通信ログ分析が注目されている．新しいマルウェアは日々大量に作成されているものの，大半は既知のマルウェアのソースコードの一部を再利用して作成されている [2]．よって，機械学習を用いれば，シグネチャで検知できないものでも，既知のマルウェアの通信パターンとの類似性を捉えて検知できる可能性がある．

機械学習を用いてマルウェア感染や悪性コンテンツの有無を二値分類する研究は多数報告されている [3–9]．これらの研究では，分類性能を比較するため様々な性能指標

[1] NTT セキュアプラットフォーム研究所
NTT Secure Platform Laboratories
[2] NTT コミュニケーション科学基礎研究所
NTT Communication Science Laboratories

を用いているが，通信ログ分析のように，大量の正常ログ（良性ログ）の中に少量のマルウェア由来の通信（悪性ログ）を含む不均衡なデータを取り扱う際，いくつかの性能指標は不適となる．例えば，99 個の良性ログと 1 個の悪性ログからなるテストデータを accuracy で評価した場合を考える．仮に機械学習が全てのログを良性と判定したとすると，悪性ログを検知していないにも関わらず，accuracy は 99%と高い数値になる．一方，このような不均衡性に対応できる性能指標として，area under the curve （AUC）がある．AUC は，true positive rate（TPR）を縦軸，false positive rate（FPR）を横軸としてプロットした receiver operating characteristic（ROC）曲線の下側の面積に相当し，正例と負例の両方の誤分類の割合を考慮しているため，不均衡なデータでもうまく性能指標として機能する．

しかし，セキュリティの実運用では AUC 全体ではなく，FPR が小さい（例：0.1%以下）領域での AUC が重要となる．実運用では怪しい通信ログが検知されると，ネットワーク管理者が当該ログを手動で分析して，攻撃の有無を最終判断しているため，FPR が大きいとネットワーク管理者の負担になる．実際，いくつかのセキュリティログ分析の先行研究 [3–6] では，この点に着目して，FPR が小さくなるように閾値を調整した際の TPR を性能指標として用いている．そこで，本研究では任意の FPR 領域の AUC（以降 pAUC と呼ぶ）を最大化する手法について検討する．

pAUC 最大化手法についてはいくつかの先行研究が存在する [10–13]．ただし，これらの研究では分類器によって付与された学習データのスコアに引き分けがないことを仮定している．しかし，通信ログ分析においては，ロードバランサ等の影響で，通信ログを特徴ベクトル化した際に引き分けが生じる場合がある（3 章参照）．そこで，本稿ではスコアの引き分けを考慮して，pAUC を最大化する教師あり学習法（pAUCBoost）を提案する．スコアの引き分けを考慮することで，pAUCBoost は従来法 [10–13] に比べてより正確に pAUC を最大化するように学習できるため，許容する FPR を定めて運用した場合，従来法では検知できなかった悪性ログを検知できる可能性がある．また，pAUCBoost は線形モデルを用いているため，各特徴量の係数（寄与率）を算出することで，どの特徴量がどの程度分類に寄与しているのかを可視化できる．実運用ではネットワーク管理者が人手で分析して攻撃の有無を最終判断するため，分類理由がわかる手法が望まれている．

本論文の主な貢献は以下の 3 つである．

- スコアが引き分けとなる場合を考慮した新たな pAUC 最大化手法を提案した．
- 著者らの知る限りにおいて，セキュリティログ分析の課題に対して，pAUC 最大化手法を始めて適用した．
- 実在の大企業網から取得した proxy ログを用いて，pAUCBoost と従来法の比較を行い，優位性を示した．

2. pAUC 最大化学習

本章では，AUC と pAUC の定義および計算例，既存の pAUC 最大化アルゴリズムについて記す．

2.1 AUC & pAUC

本稿では，通信ログが悪性（“+” クラス）か良性（“−” クラス）かを分類する二値分類問題を考える．悪性のデータセット $S^+ = \{(\boldsymbol{x}_1^+, y_1^+), (\boldsymbol{x}_2^+, y_2^+), \cdots, (\boldsymbol{x}_m^+, y_m^+)\}$ と，良性のデータセット $S^- = \{(\boldsymbol{x}_1^-, y_1^-), (\boldsymbol{x}_2^-, y_2^-), \cdots, (\boldsymbol{x}_n^-, y_n^-)\}$ が与えられたとする．ここで，$\boldsymbol{x}_p \in \mathbb{R}^D$ は p 番目のデータ点の特徴ベクトル，$y_p \in \{+, -\}$ をそのクラスとする．また，$\boldsymbol{w}$ を二値分類器のパラメータベクトル，$t \in \mathbb{R}$ を閾値，$f(\boldsymbol{x}; \boldsymbol{w})$ を $\boldsymbol{w}$ で定められるスコア関数とし，データ点 p に対して $f(\boldsymbol{x}_p; \boldsymbol{w}) > t$ となるならば悪性，$f(\boldsymbol{x}_p; \boldsymbol{w}) < t$ となるならば良性と判定されるものとする．$\boldsymbol{P}$ を確率とすると，TPR，FPR，AUC は以下のように定義される [10]．

$$\begin{aligned}
&\mathrm{TPR}_f(t) = \boldsymbol{P}[f(\boldsymbol{x}^+; \boldsymbol{w}) > t], \\
&\mathrm{FPR}_f(t) = \boldsymbol{P}[f(\boldsymbol{x}^-; \boldsymbol{w}) > t], \\
&\mathrm{AUC}_f = \int_0^1 \mathrm{TPR}_f(\mathrm{FPR}_f^{-1}(u))du. \qquad (1)
\end{aligned}$$

ここで，$\mathrm{FPR}_f^{-1}(u) = \inf\{t \in \mathbb{R} | \mathrm{FPR}_f(t) \leq u\}$ とした．

分類器によって与えられたスコアで等しくなるものが存在しないと仮定すると，経験分布による近似を行った際の TPR，FPR，AUC は以下のように書ける．

$$\begin{aligned}
&\widehat{\mathrm{TPR}}_f(t) = \frac{1}{m}\sum_{i=1}^{m} I(f(\boldsymbol{x}_i^+; \boldsymbol{w}) > t), \\
&\widehat{\mathrm{FPR}}_f(t) = \frac{1}{n}\sum_{j=1}^{n} I(f(\boldsymbol{x}_j^-; \boldsymbol{w}) > t), \\
&\widehat{\mathrm{AUC}}_f = \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n} I(f(\boldsymbol{x}_i^+; \boldsymbol{w}) > f(\boldsymbol{x}_j^-; \boldsymbol{w})). \quad (2)
\end{aligned}$$

ここで，$I(z)$ $(\forall z \in \mathbb{R})$ はヘビサイドのステップ関数であり，z が真であれば $I(z) = 1$，偽であれば $I(z) = 0$ を返す．

次に，pAUC の定義について示す．図 1 に示すように，pAUC は AUC の一部分に相当する領域である．任意の FPR 区間 $[\alpha, \beta]$ $(0 \leq \alpha < \beta \leq 1)$ における pAUC は，

$$\mathrm{pAUC}_f(\alpha, \beta) = \frac{1}{\beta - \alpha}\int_\alpha^\beta \mathrm{TPR}_f(\mathrm{FPR}_f^{-1}(u))du, \quad (3)$$

となる．同様に，経験分布による近似を行うと [11,12]，

$$\begin{aligned}
&\widehat{\mathrm{pAUC}}_f(\alpha, \beta) \\
&= \frac{1}{mn(\beta - \alpha)}\sum_{i=1}^{m}\Big[(j_\alpha - n\alpha)\cdot I(f(\boldsymbol{x}_i^+; \boldsymbol{w}) > f(\boldsymbol{x}_{(j_\alpha)}^-; \boldsymbol{w})) \\
&\quad + \sum_{j=j_\alpha+1}^{j_\beta} I(f(\boldsymbol{x}_i^+; \boldsymbol{w}) > f(\boldsymbol{x}_{(j)}^-; \boldsymbol{w})) \\
&\quad + (n\beta - j_\beta)\cdot I(f(\boldsymbol{x}_i^+; \boldsymbol{w}) > f(\boldsymbol{x}_{(j_\beta+1)}^-; \boldsymbol{w}))\Big], \qquad (4)
\end{aligned}$$

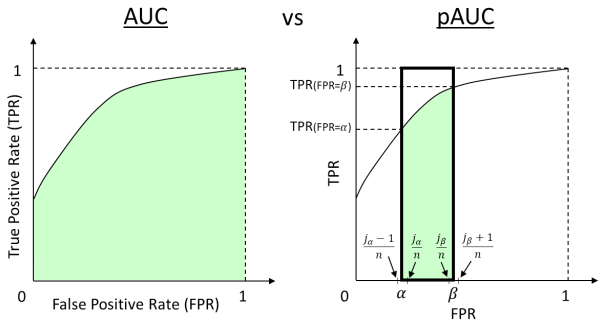

図 1　AUC と pAUC の違い．pAUC は AUC の一部分に相当する．x 軸上の各種記号は式 (4) を参照のこと．$\mathrm{TPR}_{\mathrm{FPR}=*}$（$* = \{\alpha, \beta\}$）は FPR=$*$ となるよう閾値を調整した際の TPR である．右図において，$[\alpha, \beta]$ 区間の pAUC は緑色部分の面積を太枠で囲った長方形の面積で割ったものに相当する．

となる．ここで，$j_\alpha = \lceil n\alpha \rceil$，$j_\beta = \lfloor n\beta \rfloor$ とした．記号 $\lceil * \rceil$ は $*$ 以上の最小の整数，$\lfloor * \rfloor$ は $*$ 以下の最大の整数を意味している．また，$\boldsymbol{x}^-_{(j)}$ はスコア関数 f を基に算出されたスコアのうち，上位 j 番目の良性ログを表す．なお，式 (3) と式 (4) は，図 1 右側の実線で囲った四角部分の面積で除することで，0 から 1 の値をとるように正規化されている．

式 (3) と式 (4) において，$\alpha = 0$，$\beta = 1$ のようにとると，式 (3) は式 (1) に，式 (4) は式 (2) にそれぞれ一致することに留意されたい．

2.2　AUC & pAUC の計算例

例として，5 つの悪性ログ（+ クラス）と 4 つの良性ログ（− クラス）からなるラベル付きのデータを学習した場合を考える．このとき，2 つの機械学習を行った結果，図 2 の右側のようなスコア関数 f_1 と f_2 および，スコアのランキング表が得られたとする（例えば，f_1 は Support Vector Machine（SVM），f_2 は Logistic Regression（LR）による学習で得られたスコア関数，と考える）．ランキング表の赤は悪性，青は良性のログを意味している．また，スコアは 0 から 1 の値をとり，1 に近ければ悪性の可能性が高いログ，0 に近ければ良性の可能性が高いログと学習されているものとする．

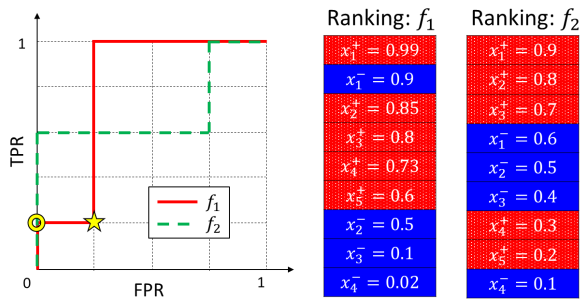

図 2　ROC 曲線の例．f_1 と f_2 は独立な 2 パターンの例である．右側はスコア関数 f_1 もしくは f_2 から得られたスコアで，左側はそれらのスコアを基に描いた ROC 曲線である．

図 2 の左側はスコア関数 f_1，f_2 それぞれによって算出されたスコアを基に，TPR，FPR を計算し，ROC 曲線を描いたものである．参考のため，スコア関数が f_1 の場合の ROC 曲線の描き方について説明する．まず閾値を 0.9 から 0.99 の間の数字に設定し，その閾値を超えた場合を悪性，下回った場合を良性だと分類されるとする．このとき，x_1^+ は悪性，$x_2^+ \sim x_5^+$ および $x_1^- \sim x_4^-$ は良性と判定される．このとき，TPR（悪性のログのうち，正しく悪性と判定できた割合）は 1/5，FPR（悪性と判定したが，良性だったものの割合）は 0 となる（これらは式 (1)，(2) からも得られる）．したがって，左図の二重丸のようにプロットできる．同様に，閾値を 0.85 から 0.9 の間に設定した場合，TPR は 1/5，FPR は 1/4 となるので，左図の星のようにプロットできる．これを繰り返してプロットした点を連結したものが ROC 曲線となる．AUC は ROC 曲線の下側の面積となるので，スコア関数 f_1 の場合は $16/20 = 0.8$，f_2 の場合は $14/20 = 0.7$ となる（これらは式 (1)，(2) からも得られる）．また，pAUC は ROC 曲線の一部の下側の面積を正規化したものなので，例えば FPR が $[0, 0.1]$ となる区間に着目すると，式 (3)，(4) より，f_1 の場合は $0.02/0.1 = 0.2$，f_2 の場合は $0.06/0.1 = 0.6$ となる．

この例の場合，AUC で比較すると，スコア関数 f_1 は f_2 よりも良い結果となるが，$[0, 0.1]$ 区間の pAUC で比較すると f_2 の方が良い結果が得られている．実際の通信ログ分析においては，低 FPR 条件下での TPR が重視されるため，f_2 の方がより理想的なスコア関数と言える．多くの研究では AUC を用いて性能比較を行っているが，このように AUC は高くても，実運用で重視される pAUC で比較すると必ずしも高性能ではない事例が存在する．本研究は，AUC 全体ではなく，低 FPR 領域の pAUC に着目して，これを最大化するアルゴリズムの提案を行う．

2.3　既存の pAUC 最大化アルゴリズム

pAUC 最大化に関する先行研究は数が限られているものの，いくつか存在する．ここではそれらの概略を記す．

一般に，教師あり学習の学習とは，目的関数を設計し，その目的関数が最大もしくは最小となるようなスコア関数を決定することを言う．多くの場合，その目的関数は accuracy を最大化するように設計されており，多くの書物や著名な機械学習ライブラリ（scikit-learn [14] など）では accuracy を最大化する目的関数が使用されている．

Dodd ら [10] は，目的関数として式（4）を用いることで pAUC を最大化するような学習法を提案している．ただし，式（4）は非凸な非線形関数であるため，計算コストが大きい点が課題である．Narasimhan らは，線形カーネルを用いた SVM の場合に限定して，SVM Struct を用いる [11] もしくは上界に制約を加える [12] ことで，式（4）を最大化する計算コストを抑える手法を提案している．

3. 課題

従来の pAUC 最大化手法 [10–13] では，式（4）を目的関数の一部に用いているが，式（4）は分類器によって与えられたスコアで引き分けになるものが存在しないことを仮定していた．仮にスコアに引き分けが生じると，式（2）と式（4）は，本来の AUC，pAUC の値との乖離が大きくなるため，これらを目的関数の一部として用いることは不適切である．図 3 にスコアに引き分けがある場合の例を示す．引き分けが生じた箇所の ROC 曲線は斜めの線となる．

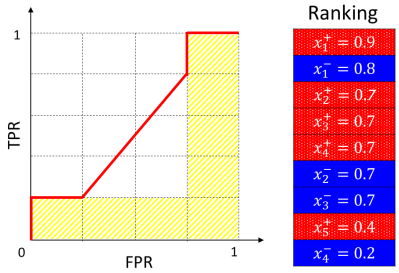

図 3 引き分けありの ROC 曲線．スコア 0.7 で引き分けている．

このとき，本来の AUC は $11/20 = 0.55$ であるが，式（2）や式（4）（$\alpha = 0$，$\beta = 1$）で計算すると，黄色の斜線で囲った領域の面積，つまり AUC$= 8/20 = 0.4$ と異なった結果が得られる（pAUC の場合も同様）．スコアに引き分けがあまり生じないタスクの場合，これは問題とならないが，通信ログ分析においては，学習の際にスコアの引き分けが生じうる事例が多数存在する．

Log	Destination IP Address	URL
Log1	192.0.2.1	http://www.example.com/RD.html
Log2	203.0.113.1	http://www.example.com/RD.html

図 4 スコアに引き分けが生じる通信ログの例．

簡単な例として，図 4 のような通信ログを考える．いくつかの著名な web サイトでは，ロードバランサなどで，URL や宛先ポート番号などが全く同じだが，負荷分散や運用上の効率化を目的として宛先 IP アドレスが複数存在している場合がある．この場合，宛先 IP アドレス以外の特徴量は Log1 と Log2 で全く同じものとなる．また，宛先 IP アドレスを特徴ベクトル化する際，例えば 192.0.2.1 と 192.0.2.2 は見た目は似た数字であっても，全く別の通信先であることも多いため，One-hot ベクトルで特徴ベクトル化することが多い．よって，仮に学習データの中で宛先 IP アドレスが 192.0.2.1，203.0.113.1 となる通信ログが 1 つずつしか存在せず，Log1，Log2 共に良性のログとして学習されると，特徴量 192.0.2.1 と 203.0.113.1 に付与される重み（特徴量の係数で寄与率を意味する）は等しくなるため，Log1，Log2 のスコアに引き分けが生じてしまう．

4. pAUCBoost

本研究では，3 章で記した課題に対して，スコアに引き分けがある場合でも，うまく pAUC を最大化できる手法を提案する．本章では提案法について記す．なお，以下簡略化のため，$f(\boldsymbol{x}_i^+; \boldsymbol{w})$ を $f(\boldsymbol{x}_i^+)$ のように略記する．

4.1 厳密な pAUC の定義

2.1 節では，分類器によって与えられたスコアで等しくなるものが存在しないと仮定していた．本節では，スコアが等しいものが存在する場合にも対応した pAUC を厳密に定義する．以下では，$[\alpha, j_\alpha/n]$ 区間，$[j_\alpha/n, j_\beta/n]$ 区間，$[j_\beta/n, \beta]$ 区間に分けて pAUC の定義を行う．

図 5 は，$[\alpha, j_\alpha/n]$ 区間の pAUC の場合を表している．図中の斜め線はスコアが引き分けがあった場合の ROC 曲線をイメージしている．① ～ ④ は TPR, FPR の定義から求まり，⑤，⑥ は ① ～ ④ から得られる．$[\alpha, j_\alpha/n]$ 区間の pAUC に相当する，台形部分の面積を求めると，厳密な pAUC の計算式は以下のようになる（$[j_\alpha/n, j_\beta/n]$ 区間，$[j_\beta/n, \beta]$ 区間の場合も同様）．

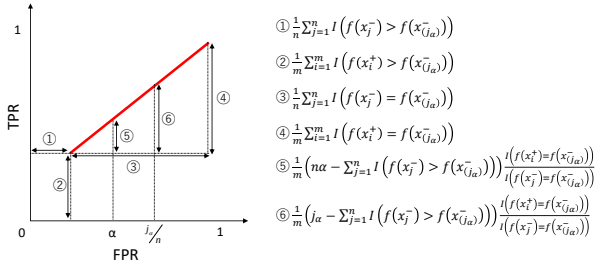

図 5 $[\alpha, j_\alpha/n]$ 区間の場合．

$[\alpha, j_\alpha/n]$ 区間（$j_\alpha \neq 0$ のとき）

$$\frac{1}{mn(\beta-\alpha)}(j_\alpha - n\alpha)\left[\sum_{i=1}^{m} I(f(\boldsymbol{x}_i^+) > f(\boldsymbol{x}_{(j_\alpha)}^-)) - \sum_{j=1}^{n} I(f(\boldsymbol{x}_j^-) > f(\boldsymbol{x}_{(j_\alpha)}^-))\frac{\sum_{i=1}^{m} I(f(\boldsymbol{x}_i^+) = f(\boldsymbol{x}_{(j_\alpha)}^-))}{\sum_{j=1}^{n} I(f(\boldsymbol{x}_j^-) = f(\boldsymbol{x}_{(j_\alpha)}^-))} + \frac{1}{2}(j_\alpha + n\alpha)\frac{\sum_{i=1}^{m} I(f(\boldsymbol{x}_i^+) = f(\boldsymbol{x}_{(j_\alpha)}^-))}{\sum_{j=1}^{n} I(f(\boldsymbol{x}_j^-) = f(\boldsymbol{x}_{(j_\alpha)}^-))}\right]. \quad (5)$$

$[j_\alpha/n, j_\beta/n]$ 区間

$$\frac{1}{mn(\beta-\alpha)}\sum_{i=1}^{m}\sum_{j=j_{\alpha+1}}^{j_\beta}\{I(f(\boldsymbol{x}_i^+) > f(\boldsymbol{x}_{(j)}^-)) + \frac{1}{2}I(f(\boldsymbol{x}_i^+) = f(\boldsymbol{x}_{(j)}^-))\}. \quad (6)$$

$[j_\beta/n, \beta]$ 区間（$j_\beta \neq n$ のとき）

$$\frac{1}{mn(\beta-\alpha)}(n\beta-j_\beta)\left[\sum_{i=1}^{m} I(f(\boldsymbol{x}_i^+) > f(\boldsymbol{x}_{(j_\beta+1)}^-)) - \sum_{j=1}^{n} I(f(\boldsymbol{x}_j^-) > f(\boldsymbol{x}_{(j_\beta+1)}^-)) \frac{\sum_{i=1}^{m} I(f(\boldsymbol{x}_i^+) = f(\boldsymbol{x}_{(j_\beta+1)}^-))}{\sum_{j=1}^{n} I(f(\boldsymbol{x}_j^-) = f(\boldsymbol{x}_{(j_\beta+1)}^-))} + \frac{1}{2}(n\beta+j_\beta)\frac{\sum_{i=1}^{m} I(f(\boldsymbol{x}_i^+) = f(\boldsymbol{x}_{(j_\beta+1)}^-))}{\sum_{j=1}^{n} I(f(\boldsymbol{x}_j^-) = f(\boldsymbol{x}_{(j_\beta+1)}^-))}\right]. \quad (7)$$

なお，最大値が 1 になるように正規化を行うため，$\beta-\alpha$ で除している．$j_\alpha = 0$ のときは式 (5)$= 0$，$j_\beta = n$ のときは式 (7)$= 0$ となる．ちなみに，$j_\alpha \neq 0$ もしくは $j_\beta \neq n$ の場合，式 (5)，(7) の分数部分の分母は 0 にならず，それぞれ 1 以上の整数となることに留意されたい．

4.2 定式化

本節では，式 (5)$\sim$ (7) を目的関数の一部として活用することで任意の $[\alpha, \beta]$ 区間の pAUC を最大化する手法について述べる．まず，ヘビサイドのステップ関数 I は微分不可能であるため，以下のように，不等式部分をロジスティックシグモイド関数 σ で，等号部分を最大値を 1 にした指数関数 ν で近似する（$\boldsymbol{x}_1$ と $\boldsymbol{x}_2$ は任意のベクトル）．

$$\sigma(\boldsymbol{x}_1, \boldsymbol{x}_2) = [1 + \exp\{-(f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2))\}]^{-1}, \quad (8)$$

$$\nu(\boldsymbol{x}_1, \boldsymbol{x}_2) = \exp\left(-\frac{(f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2))^2}{2\varsigma^2}\right). \quad (9)$$

ς は分散を意味し，ハイパーパラメータである．

これらの近似を適用すると，式 (5)$\sim$(7) は，

$[\alpha, j_\alpha/n]$ 区間（$j_\alpha \neq 0$ のとき）

$$\frac{1}{mn(\beta-\alpha)}(j_\alpha - n\alpha)\left[\sum_{i=1}^{m}\sigma(\boldsymbol{x}_i^+, \boldsymbol{x}_{(j_\alpha)}^-) - \sum_{j=1}^{n}\sigma(\boldsymbol{x}_j^-, \boldsymbol{x}_{(j_\alpha)}^-)\frac{\sum_{i=1}^{m}\nu(\boldsymbol{x}_i^+, \boldsymbol{x}_{(j_\alpha)}^-)}{\sum_{j=1}^{n}\nu(\boldsymbol{x}_j^-, \boldsymbol{x}_{(j_\alpha)}^-)} + \frac{1}{2}(j_\alpha + n\alpha)\frac{\sum_{i=1}^{m}\nu(\boldsymbol{x}_i^+, \boldsymbol{x}_{(j_\alpha)}^-)}{\sum_{j=1}^{n}\nu(\boldsymbol{x}_j^-, \boldsymbol{x}_{(j_\alpha)}^-)}\right], \quad (10)$$

$[j_\alpha/n, j_\beta/n]$ 区間

$$\frac{1}{mn(\beta-\alpha)}\sum_{i=1}^{m}\sum_{j=j_{\alpha+1}}^{j_\beta}\sigma(\boldsymbol{x}_i^+, \boldsymbol{x}_{(j)}^-), \quad (11)$$

$[j_\beta/n, \beta]$ 区間（$j_\beta \neq n$ のとき）

$$\frac{1}{mn(\beta-\alpha)}(n\beta - j_\beta)\left[\sum_{i=1}^{m}\sigma(\boldsymbol{x}_i^+, \boldsymbol{x}_{(j_\beta+1)}^-) - \sum_{j=1}^{n}\sigma(\boldsymbol{x}_j^-, \boldsymbol{x}_{(j_\beta+1)}^-)\frac{\sum_{i=1}^{m}\nu(\boldsymbol{x}_i^+, \boldsymbol{x}_{(j_\beta+1)}^-)}{\sum_{j=1}^{n}\nu(\boldsymbol{x}_j^-, \boldsymbol{x}_{(j_\beta+1)}^-)} + \frac{1}{2}(n\beta + j_\beta)\frac{\sum_{i=1}^{m}\nu(\boldsymbol{x}_i^+, \boldsymbol{x}_{(j_\beta+1)}^-)}{\sum_{j=1}^{n}\nu(\boldsymbol{x}_j^-, \boldsymbol{x}_{(j_\beta+1)}^-)}\right], \quad (12)$$

となる．また，簡単化のため，$j = j_\alpha$，$j = J$（J は $j_\alpha + 1$ 以上 j_β 以下の整数とする），$j = j_\beta + 1$，それぞれの場合に対して，下式を満たす $s(\boldsymbol{x}_i^+, \boldsymbol{x}_{(j)}^-)$ を導入する．

$$\begin{aligned} s(\boldsymbol{x}_i^+, \boldsymbol{x}_{(j_\alpha)}^-) &= (j_\alpha - n\alpha)\left[\frac{1}{2}(j_\alpha + n\alpha)\frac{\nu(\boldsymbol{x}_i^+, \boldsymbol{x}_{(j_\alpha)}^-)}{\sum_{j=1}^{n}\nu(\boldsymbol{x}_j^-, \boldsymbol{x}_{(j_\alpha)}^-)} + \sigma(\boldsymbol{x}_i^+, \boldsymbol{x}_{(j_\alpha)}^-) - \nu(\boldsymbol{x}_i^+, \boldsymbol{x}_{(j_\alpha)}^-)\frac{\sum_{j=1}^{n}\sigma(\boldsymbol{x}_j^-, \boldsymbol{x}_{(j_\alpha)}^-)}{\sum_{j=1}^{n}\nu(\boldsymbol{x}_j^-, \boldsymbol{x}_{(j_\alpha)}^-)}\right], \\ s(\boldsymbol{x}_i^+, \boldsymbol{x}_{(J)}^-) &= \sigma(\boldsymbol{x}_i^+, \boldsymbol{x}_{(J)}^-), \quad (13) \\ s(\boldsymbol{x}_i^+, \boldsymbol{x}_{(j_\beta+1)}^-) &= (n\beta - j_\beta)\left[\frac{1}{2}(n\beta + j_\beta)\frac{\nu(\boldsymbol{x}_i^+, \boldsymbol{x}_{(j_\beta+1)}^-)}{\sum_{j=1}^{n}\nu(\boldsymbol{x}_j^-, \boldsymbol{x}_{(j_\beta+1)}^-)} + \sigma(\boldsymbol{x}_i^+, \boldsymbol{x}_{(j_\beta+1)}^-) - \nu(\boldsymbol{x}_i^+, \boldsymbol{x}_{(j_\beta+1)}^-)\frac{\sum_{j=1}^{n}\sigma(\boldsymbol{x}_j^-, \boldsymbol{x}_{(j_\beta+1)}^-)}{\sum_{j=1}^{n}\nu(\boldsymbol{x}_j^-, \boldsymbol{x}_{(j_\beta+1)}^-)}\right]. \end{aligned}$$

式 (10)$\sim$(12) の対数をとって，過学習を避けるために正則化項を加えると，pAUC を最大化する目的関数 E は，

$$E(\boldsymbol{w}) = \log\left(\sum_{i=1}^{m}\sum_{j=j_\alpha}^{j_\beta+1} s(\boldsymbol{x}_i^+, \boldsymbol{x}_{(j)}^-)\right) - CR(\boldsymbol{w}), \quad (14)$$

となる．ここで，$C \in \mathbb{R}$ はハイパーパラメータで，$R(\boldsymbol{w})$ は正則化関数である．正規化関数の代表的なものとして，L_1 正則化（$||\boldsymbol{w}||$），L_2 正則化（$||\boldsymbol{w}||^2$）が挙げられる．なお，定数項は最適化計算に影響しないため消去した．

ただし，Eq. (14) は $\boldsymbol{w}$ に関して凸でないため，計算が難しい．そこで，EM アルゴリズムに倣って，$E(\boldsymbol{w})$ の下界を最大化することで，初期値周辺で $E(\boldsymbol{w})$ を最大化する $\boldsymbol{w}$ を推定する．ここで，$E_{LB}(\boldsymbol{w}, q_{i(j)})$ を $E(\boldsymbol{w}) \geq E_{LB}(\boldsymbol{w}, q_{i(j)})$ を満たす $E(\boldsymbol{w})$ の下界と定義する．$q_{i(j)}$ は $q_{i(j)} \geq 0$ かつ $\sum_{i=1}^{m}\sum_{j=j_\alpha}^{j_\beta+1} q_{i(j)} = 1$ を満たすものとした．すると，以下の不等式が成立する（紙面の都合上証明略）．

$$\log\left(\sum_{i=1}^{m}\sum_{j=j_\alpha}^{j_\beta+1} s(\boldsymbol{x}_i^+, \boldsymbol{x}_{(j)}^-)\right) \geq \sum_{i=1}^{m}\sum_{j=j_\alpha}^{j_\beta+1} q_{i(j)} \log s(\boldsymbol{x}_i^+, \boldsymbol{x}_{(j)}^-) - \sum_{i=1}^{m}\sum_{j=j_\alpha}^{j_\beta+1} q_{i(j)} \log q_{i(j)}. \quad (15)$$

したがって，式 (14) の下界は以下のような凸関数になる．

$$E_{LB} = \sum_{i=1}^{m}\sum_{j=j_\alpha}^{j_\beta+1} q_{i(j)} \log s(\boldsymbol{x}_i^+, \boldsymbol{x}_{(j)}^-) - \sum_{i=1}^{m}\sum_{j=j_\alpha}^{j_\beta+1} q_{i(j)} \log q_{i(j)} - CR(\boldsymbol{w}). \quad (16)$$

なお，式 (16) は $j = j_\alpha \sim j_\beta + 1$ の範囲内の良性ログ（“–” クラス）のみが関係しており，範囲外の良性ログは無視されていることに注意されたい．

ここで，$\hat{\boldsymbol{w}}$ を前ステップの解とする．$\sum_{ij} q_{i(j)} = 1$ の拘束条件下でラグランジュの未定乗数法を用いると（紙面上の都合で証明は省略する），E_{LB} を最大化する $q_{i(j)}$ は以下のように得られる．

$$\hat{q}_{i(j)} = \frac{s(\boldsymbol{x}_i^+, \boldsymbol{x}_{(j)}^-)}{\sum_{i'=1}^{m}\sum_{j'=j_\alpha}^{j_\beta+1} s(\boldsymbol{x}_{i'}^+, \boldsymbol{x}_{(j')}^-)}. \quad (17)$$

以上より，式 (17) の $\hat{q}_{i(j)}$ と $\hat{\boldsymbol{w}}$ を交互最適化しながら，目的関数式 (16) を最大化することで，pAUC を最大化する最適なパラメータベクトル $\hat{\boldsymbol{w}}$ を得ることができる．

4.3 pAUCBoost のアルゴリズム

まず，スコア関数の取り方について議論する．pAUCBoost においてスコア関数は線形でも非線形な形を取っても先章までの数式は成立する．Ueda [13] らは非線形なスコア関数を用いた場合の pAUC 最大化について議論している．ただし，セキュリティログ分析においては最終的な判断はネットワーク管理者が人手でログを分析して行っているため，どの特徴量が効いて良性/悪性と判断されたかは重要な情報となるため，本研究ではスコア関数 $f(\boldsymbol{x})$ は $f(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}$ のように線形のスコア関数を用いることとした．線形のスコア関数であれば，各特徴量に対応したパラメータベクトル $\boldsymbol{w}$ の値を見ることで，どの特徴量が最終的な判断にどれだけ影響しているかを調べることができる．

以上より，提案法（pAUCBoost）のアルゴリズムは以下のようになる．

Algorithm 1 pAUCBoost アルゴリズム

Require: Training datasets S^+ and S^-, scoring function $f(\boldsymbol{x}; \hat{\boldsymbol{w}})$, regularization function $R(\hat{\boldsymbol{w}})$, and hyperparameter C and ς.
Ensure: $\hat{\boldsymbol{w}}$.
1: Initialize $\hat{\boldsymbol{w}}$.
2: **while** not converged **do**
3: Update $\hat{q}_{i(j)}$ in Eq. (17).
4: Update $\hat{\boldsymbol{w}}$ by maximizing Eq. (16).
5: **end while**

なお，本研究では，過学習を避けるため，L_2 正則化を用いた．また，式 (16) を最適化する際には，L-BFGS アルゴリズム [15] を用いた．L-BFGS アルゴリズムは準ニュートン法の最適化アルゴリズムで様々な機械学習法で活用事例がある．Algorithm 1 における収束判定条件としては，E_{new} を更新後の E_{LB}，E_{old} を更新前の E_{LB} とすると，$||1 - E_{new}/E_{old}|| \leq \epsilon$ が満たされた場合，もしくは十分な繰り返し計算を行った場合（本稿では 100 回に設定）に収束した，と判定することとした．ここでの ϵ は小さい値を意味しており，本稿では 0.001 に設定した．

[注意点 1] pAUCBoost は収束条件を満たすまで，式 (16) と (17) の更新を続ける必要がある（Algorithm 1 中の while 文）．これは，各イタレーションで得られた $\hat{\boldsymbol{w}}$ によって，スコア関数 $f(\boldsymbol{x}; \hat{\boldsymbol{w}})$ および目的関数の式 (16) が更新されるためである．

[注意点 2] pAUCBoost は任意の $[\alpha, \beta]$ 区間の pAUC を最大化するように学習するため，AUC 全体を最大化する場合（$\alpha = 0$，$\beta = 1$ とすればよい）でも成立する．

5. データセット

本稿では，proxy ログが良性（正規の通信）か悪性（マルウェア由来の通信）かを判定する問題について，提案法と既存手法との精度比較を行う．proxy ログには，送信元/宛先 IP アドレス，送信元/宛先ポート番号，送信元/宛先パケット数，リクエスト/レスポンスのバイト数，URL，ユーザエージェント，ステータスコード，HTTP メソッド，タイムスタンプなどの情報が含まれている．多数のマルウェアがドライブバイダウンロードや C&C サーバへの通信の際に HTTP 通信を用いているため，本研究では proxy ログを用いて実験を行った．

良性のデータはある大企業 1 社の社内網から取得した．良性のデータセットの中に悪性の通信が紛れ込まないよう，ゲートウェイからエンドポイントまでの複数のモニタリングポイントに商用のアンチウイルス製品を多数設置し，悪性通信がないことを確認している．

悪性のデータはマルウェアの検体を VirusTotal [16] からダウンロードし，サンドボックス [17] で動的解析して得た．各検体は VirusTotal の 5 つ以上のアンチウイルスソフトから悪性と判定されたもので，各種アンチウイルスソフトによる検知傾向がランダムになるよう，日々最新のものを 5 年間に渡って一定数収集している．また，各検体は各々 SHA1 Hash が異なり，様々なマルウェアファミリを含んでいる（ESET [18] で調査したところ，4,941 検体（126,171 ログ）内に，73 種類のマルウェアファミリが含まれていた）．さらに，悪性のデータセット内に良性の通信が紛れ込まないよう，Alexa の上位 100 万位 [19] のドメインに該当するログは良性の可能性が高いとして除去した．

表 1 データセットに含まれる端末数とログ数

	良性		悪性	
	端末数	ログ数	検体数	ログ数
学習	1,094	48,869	218	5,000
検証	1,054	46,638	274	5,000
テスト	1,006	44,562	273	5,000

データセットの端末数とログ数を表 1 として示す．学習の効率化のため，全く同じ通信ログ，つまり重複しているログは削除した．また，取得日が古いものから順に，学習データ，検証データ，テストデータとした．なお，検証データはホールドアウト検証を行うために用意している．ハイパーパラメータを調整する際，一般的に交差検定などが用いられているが，交差検定を用いると，時系列的に後に発見されたマルウェアの検体のログを用いて分類器を生成し，前に発見されたマルウェアを検知する，という検証を行ってしまう．しかし，新しいマルウェアをどれだけ誤検知少なく検知できるのかが検証の趣旨であるため，本稿では時系列を考慮してホールドアウト検証を採用した．

6. 特徴量

本章では実験で用いた特徴量について記す．本稿で用いた特徴量は評価の公平性を期すため，表 2 のような広く一般に用いられているものを用いた [20, 21].

表 2 中のパスエレメントは URL のパスを “/” で区切った後の各単語を意味している．例えば，“http://www.example.com/RD/index.php” であれば，“RD” と “index.php” が該当する．AS 番号，国，市は，宛先 IP アドレスを Maxmind 社の GeoIP Lite データベース [22] に問い合わせて得られた結果を利用している．

単語ベースの特徴量をベクトル化する際には，bag-of-words モデルを用いた．各特徴量に生じる全てのパターンを 1 つの要素とみなし，その要素がログの中に存在するか否かで，1 もしくは 0 を割り当てた．図 6 に bag-of-words モデルで特徴ベクトル化した例を示す．この例では，各宛先 IP アドレスと TLD を要素として扱っている．よって，Log1 から 3 の特徴ベクトルは，Log1=$[1\ 0\ 0\ 1\ 0]^{\mathrm{T}}$，Log2=$[0\ 1\ 0\ 0\ 1]^{\mathrm{T}}$，Log3=$[0\ 0\ 1\ 0\ 1]^{\mathrm{T}}$ となる．

Log	Destination IP Address	TLD
Log1	192.0.2.5	.us
Log2	198.51.100.5	.com
Log3	203.0.113.5	.com

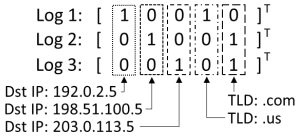

図 6 bag-of-words モデルを使った特徴ベクトル化の例

統計量ベースの特徴は，学習用データで想定される最も大きな値で割ることで正規化した．例えば，URL の全体もしくは一部の長さの特徴は，URL の長さの最大値である 2083 で割って正規化した．また，* がついた特徴量は，存在する場合は 1，存在しない場合は 0 として実装した．

表 2 特徴量の種類

単語ベース	特徴量	統計量ベース	特徴量
HTTP	FQDN	長さ	URL
	TLD		ドメイン
	パスエレメント		FQDN
	クエリキー		パス
	クエリパラメータ		クエリ
	UserAgent		ファイル名
	メソッド		拡張子
TCP/IP	宛先 IP アドレス	存在の有無	パス内の記号 *
	宛先ポート番号		URL 内の拡張子 *
GeoIP	AS 番号	数字の数	URL
	国		FQDN
	市		パス

7. 評価

proxy ログが良性か悪性かを分類する問題について，pAUCBoost と従来の教師あり学習の性能を比較する．従来の教師あり学習として，SVMpAUC [11]，L_2 正則化付きの LR，L_2 正則化 TPR の付きの線形カーネルの SVM を用いた．SVMpAUC は先行研究 [11] の pAUC 最大化学習法である．通信ログ分析においては，最終的な攻撃の有無の判断は人手で行うことが多いため，本稿ではどの特徴量がどの程度分類に寄与しているのかがわかる，線形モデルの学習アルゴリズムを比較対象として用いている．

ハイパーパラメータ（pAUCBoost では ς，C）は $10^{-10}, 10^{-9} \cdots 10^{10}$ の範囲内で，それぞれの性能指標が最大となるよう，ホールドアウト検証を用いて定めた．また，本実験では，性能指標として $\mathrm{pAUC}_{[\alpha,\beta]}$，AUC，$\mathrm{TPR}_{\mathrm{FPR=X\%}}$ を用いた．ここで，$\mathrm{pAUC}_{[\alpha,\beta]}$ は $[\alpha, \beta]$ 区間の pAUC，$\mathrm{TPR}_{\mathrm{FPR=X\%}}$ は FPR = X% となるように閾値を調整した際の TPR の値，をそれぞれ表している．$\mathrm{pAUC}_{[\alpha,\beta]}$ と $\mathrm{TPR}_{\mathrm{FPR}=\alpha}$，$\mathrm{TPR}_{\mathrm{FPR}=\beta}$ の違いについては図 1 を参照のこと．α，β の値は任意に定めても良いが，セキュリティログ分析においては低 FPR 領域の TPR が重要なので，本稿では，$[\alpha, \beta] = [0, 0.001]$，$[\alpha, \beta] = [0, 0.01]$，$[\alpha, \beta] = [0, 1]$（つまり AUC 全域）と設定した．本稿では α，β を先述のように定めたが，実運用ではネットワーク管理者の稼働とテストデータのログ数から許容できる FPR の量から決めればよい．

以上の操作を経て得た実験結果を表 3 に示す．

表 3 実験結果

Method	pAUCBoost	SVMpAUC	LR	SVM
$\mathrm{pAUC}_{[0,0.001]}$	**0.5792**	0.0650	0.5063	0.5033
$\mathrm{pAUC}_{[0,0.01]}$	**0.7781**	0.2891	0.7545	0.7587
AUC	**0.9939**	0.9703	0.9932	0.9917
$\mathrm{TPR}_{\mathrm{FPR=0.1\%}}$	**0.6706**	0.2356	0.5890	0.6115
$\mathrm{TPR}_{\mathrm{FPR=1\%}}$	**0.8702**	0.4057	0.8508	0.8607

pAUCBoost は全指標に対して，従来の教師あり学習手法に比べて良い性能を示した．pAUCBoost は指定した範囲の pAUC を最大化するように学習しているため，特に低誤検知領域では pAUC 最大化の効果が顕著に表れていることも確認できた．実運用では低誤検知領域が重視されるため，これは有意義な結果である．一方，SVMpAUC はあまりよい性能を示さなかった．これは学習データの中に分類器から与えられたスコアの引き分けが多数存在したことがその要因と考えられる．なお，本実験で用いた学習データにおいて，スコアが引き分けとなるログの数は，良性が約 25%（12,322 ログ），悪性が約 19%（967 ログ）であった．

8. 関連研究

教師あり学習を用いて通信ログから悪性のコンテンツを検知する手法は多数存在する．表 4 にいくつかの最新の研究で用いられていた性能指標についてまとめた．いくつかの研究では $\mathrm{TPR}_{\mathrm{FPR=X\%}}$ [3–6] に焦点を当てているが，我々の知る限りにおいて，直接 pAUC を最大化した研究や，pAUC を性能指標として用いている研究はなかった．

表 4 関連研究で用いられていた性能指標

性能指標/研究	[3]	[4]	[5]	[6]	[7]	[8]	[9]
$\mathrm{TPR}_{\mathrm{FPR=X\%}}$	✓	✓	✓	✓			
AUC	✓	✓		✓		✓	
Accuracy							✓
TPR	✓	✓	✓	✓	✓	✓	✓
FPR		✓	✓	✓	✓	✓	
Precision					✓		✓
F-measure		✓				✓	✓

一般に知られている教師あり学習では accuracy を最大化するように学習させているが，いくつかの研究では二値分類の性能指標として AUC に着目して，これを直接最適化する手法を提案している [23,24]．しかし，医療などいくつかの機械学習のタスクにおいては，AUC よりも低誤検知領域での TPR など，特定の FPR 領域条件下での AUC（つまり pAUC）の向上が必要とされる．いくつかの研究では [11–13]，pAUC を直接最適化する手法を提案している．Narasimhan et al. [11,12] は線形カーネルの Support Vector Machine の場合に限って pAUC を直接最適化する方法を提案している．また，Ueda et al. [13] は非線形スコア関数を用いた場合の pAUC 最大化法について述べている．これらの研究では，分類器が与えたスコアに引き分けがない場合を仮定しているが，セキュリティログ分析においては多数の引き分けが生じうるため，本研究では引き分けを考慮する手法を提案した．

9. 結論

本稿では，任意の FPR 区間における TPR を最大化する手法を提案した．先行研究では考慮していなかった，通信ログに多いスコアの引き分けを考慮している点が提案法の長所である．これにより，より正確に任意の FPR 区間における TPR を最大化することができる．通信ログ分析では，ロードバランサなどで通信内容が同じでも宛先 IP アドレスだけが異なる事例など，似たような特徴ベクトルが多くなる傾向があるため，スコアの引き分けを考慮する必要がある．また，大企業網から得た proxy ログを用いて提案法と従来法の精度比較を行い，提案法の優位性を示した．

参考文献

[1] Griffin, K., Schneider, S., Hu, X., Chiueh, T.: Automatic generation of string signatures for malware detection. RAID, vol. 5758, pp. 101-120. Springer (2009)

[2] Jang, J., Brumley, D., Venkataraman, S.: BitShred: feature hashing malware for scalable triage and semantic analysis. CCS, pp. 309-320. ACM (2011)

[3] Mirsky, Y., Doitshman, T., Elovici, Y., Shabtai, A.: Kitsune: an ensemble of autoencoders for online network intrusion detection. NDSS. Internet Society (2019)

[4] Copty, F., Danos, M., Edelstein, O., Eisner, C., Murik, D., Zeltser, B.: Accurate malware detection by extreme abstraction. ACSAC, pp. 101-111. ACM (2018)

[5] Wressnegger, C., Yamaguchi, F., Arp, D., Rieck, K.: Comprehensive analysis and detection of flash-based malware. DIMVA, vol. 9721, pp. 101-121. Springer (2016)

[6] Hendler, D., Kels, S., Rubin, A.: Detecting malicious powershell commands using deep neural networks. AsiaCCS, pp. 187-197. ACM, (2018)

[7] Pereira, M., Coleman, S., Yu, B., DeCock, M., Nascimento, A.: Dictionary extraction and detection of algorithmically generated domain names in passive dns traffic. RAID, vol. 11050, pp. 295-314. Springer (2018)

[8] Zhang, J., Jang, J., Gu, G., Stoecklin, M., Hu, X.: ErrorSesor: mining information from http error traffic for malware intelligence. RAID, vol. 11050, pp. 467-489. Springer (2018)

[9] Stergiopoulos, G., Talavari, A., Bitsikas, E., Gritzalis, D.: Automatic detection of various malicious traffic using side channel features on tcp packets. ESORICS, vol. 11098, pp. 346-362. Springer (2018)

[10] Dodd, L., Pepe, M.: Partial auc estimation and regression. Biometrics **59**(3), 614–623 (2003)

[11] Narasimhan, H., Agarwal, S.: A structural svm based approach for optimizing partial auc. ICML, vol. 28. (2013)

[12] Narasimhan, H., Agarwal, S.: SVMpAUCtight: a new support vector method for optimizing partial auc based on a tight convex upper bound. KDD, pp. 167-175. ACM (2013)

[13] Ueda, N., Fujino, A.: Partial auc maximization via nonlinear scoring functions. arXiv preprint arXiv:1806.04838v1 (2018)

[14] scikit-learn, https://scikit-learn.org/stable/

[15] Zhu, C., Byrd, R., Lu, P., Nocedal, J.: Algorithm 778: l-bfgs-b: fortran subroutines for large-scale bound-constrained optimization. TOMS **23**(4), 550–560 (1997)

[16] virustotal, https://www.virustotal.com/

[17] Aoki, K., Yagi, T., Iwamura, M., Itoh, Mitsutaka.: Controlling malware http communications in dynamic analysis system using search engine. CSS, pp.1-6. IEEE (2011)

[18] ESET Homepage, https://www.eset.com/

[19] AWS, http://s3.amazonaws.com/alexa-static/

[20] Ma, J., Saul, L., Savage, S., Voelker, G.: Learning to detect malicious urls. TIST **2**(3), 30:1–30:24 (2011)

[21] Sahoo, D., Liu, C., Hoi, C.: Malicious url detection using machine learning: a survey. arXiv preprint arXiv:1701.07179 (2017)

[22] Python, https://pypi.org/project/geoip2/

[23] Herschtal, A., Raskutti, B.: Optimising area under the roc curve using gradient descent. ICML. ACM (2004)

[24] Calders, T., Jaroszewicz, S.: Efficient auc optimization for classification. PKDD, vol. 4702, pp. 42-53. Springer (2007)