DBSJ Japanese Journal
Vol. 16-J, Article No. 6, March 2018

外れ構造検知:大規模離散値データの圧縮による集団アノマリの発見

Compression-based Discovery of Anomalous Behavior in Large-scale Categorical Data

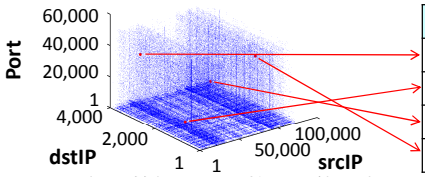
丸橋 弘治♡ 齊藤 聡美♢ 鳥居 悟♠ 武仲 正彦♣

Koji MARUHASHI Satomi SAITO
Satoru TORII Masahiko TAKENAKA

大規模な離散値データに基づき異常行動を効率的に発見することが求められている。従来の異常検知手法は稀な値やパタンを持つデータを抽出するが、離散値のとりうる値が数万以上の大規模データになると、大半の値やパタンが稀なものになる。その場合、単一のデータを抽出するのではなく、値の関連の仕方に注目し、データの集団としてのアノマリを検知することが必要となる。本研究では、従来の異常検知手法にデータ集団の考え方を取り入れた、新たなアノマリ検知手法を提案する。また、アノマリの候補となる小規模なデータ集団を大量に生成するための、新たなクラスタリング手法を提案する。さらに、実際の侵入検知ログを用いた実験により、本手法の効果を実証する。

How can we discover anomalous behaviors hidden in very large categorical data? Most of the existing anomaly detection methods identify records with rare values or patterns; however, such approaches would have difficulties with datasets containing several attributes with tens of thousands of categorical values, because the number of rare values becomes huge in such datasets. Therefore we need some additional viewpoints on the definition of anomalies other than rarity. In this work, we propose a new anomaly detection method for large-scale categorical data that can detect a group of records or values that *collectively* exhibit anomalous behaviors, i.e, *anomalous cluster*. Moreover, we develop a new algorithm to create more than thousands of candidate clusters of anomalous clusters. We report the results of empirical studies on real-world IDS logs, showing the effectiveness of our method.

♡ 正会員　株式会社富士通研究所
maruhashi.koji@jp.fujitsu.com
♢ 株式会社富士通研究所
sa.satomi@jp.fujitsu.com
♠ 株式会社富士通研究所
torii.satoru@jp.fujitsu.com
♣ 株式会社富士通研究所
ma@jp.fujitsu.com



srcIP	dstIP	Port
a.a.a.a	x.x.x.x	p1
a.a.a.a	y.y.y.y	p2
b.b.b.b	x.x.x.x	p2
b.b.b.b	y.y.y.y	p1

図 1: 侵入検知ログの数百万件以上のアラートのプロット（左）。ほとんどの送信元 IP(srcIP)、送信先 IP(dstIP)、ポート番号 (Port) は稀な値であるが、稀な値を共有しあうデータの集団（右）は、何らかの意味のある挙動を反映している可能性がある。

1 はじめに

人やモノの挙動が反映された大規模データから、異常な行動を発見することが強く求められている。例えば、ネットワークシステムへの不正な侵入を知らせるアラート（侵入検知ログ）は、送信元 IP、送信先 IP、ポート番号といった数万種類の離散値を含むデータである（図 1）。企業のセキュリティ担当者は、大半を占める日常的なアラートの中から、従来知られていない類の攻撃を発見し、早急に対策を行う必要がある。そのため、大規模な離散値データの中から、他と異なる異常なデータ（アノマリ）を抽出するだけでなく、対策を検討するため、異常の理由が容易に分析可能な技術が求められる。このような技術として、データ圧縮を利用したアノマリ検知手法が提案されている [13, 2]。これらの手法は、値やパタンにコードを与えるコード表を生成する。このとき、データ全体を効率的に圧縮できるよう、頻出する値・パタンほど短いコードを与える。これにより、頻出パタンで記述できないために記述長が長いデータをアノマリとして抽出すると共に、異常の理由となるパタンを容易に分析することが可能となる。

しかし、数百万件以上のデータを含む大規模離散値データでは、ほとんどの値・パタンが稀になってしまう。このようなデータにおいては、稀な値・パタンを含む単一のデータを抽出するのではなく、互いに何らかの意味を持って関連しあっている、もしくは互いの関連の仕方が他と異なるデータの集団を、アノマリとして検知することが必要となる。例えば、図 1 右に示したデータの集団は、それぞれの IP アドレスやポート番号は 1 回か 2 回しか使われない稀な値であるが、データ間で値を共有し互いに関連しあっていることから、単なるノイズではなく、何らかの注目すべき挙動と関連している可能性がある。実際、巧妙な攻撃が残した少ない痕跡を大量の侵入検知ログから検知することは容易ではないが、表 6 に示すような小規模なデータ集団を抽出することにより発見できる可能性がある。

本研究では、データ圧縮を利用したアノマリ検知手法に、データ集団の考え方を取り入れた新たな手法を提案する。提案手法では、データ集団内での値の共有のされ方を反映した、データ集団別のコード表を導入する。そして、従来手法を拡張し、頻出パタンで記述できないために記述長が長いだけでなく、データ集団内での値の共有の度合いも考慮に入れて、データ集団としてのアノマリを抽出する。

また、離散値データの従来のクラスタリング手法 [1] は、数万種類の離散値を含む大規模データに対し、アノマリの候補となる数千個以上の小規模なデータ集団を効率的に生成することはできない。さらに、ほとんどの従来手法ではデータ集団の数を指定する必要があるが、数千個以上のデータ集団の数を予め知ることは困難である。本研究では、離散値データの新たなクラスタリング手法も提案する。

本研究の貢献は、以下のとおりである。

- データ圧縮を利用した、新たなアノマリ検知手法を提案する。提案手法は、データ集団別のコード表を用いて、頻出パタンで記述できないだけでなく、データ集団内での値の共有の度合いも考慮に入れて、アノマリであるデータ集団を抽出する。

- 大規模離散値データから大量の小規模なデータ集団を効率的に生成する、新たなクラスタリング手法 *DescCat* を提案する。DescCat は、最小記述長原理 (MDL) に基づき、データ集団別のコード表の記述長と、データ全体の記述長の合計を最小化する。また、予めデータ集団の数を与える必要はない。
- 侵入検知ログを用い、多くのタイプの攻撃を従来手法より高精度に検知できることを示す。さらに、DescCat は、大規模データセットを従来手法より効率的に圧縮することを示す。

2 関連研究

2.1 離散値データからのアノマリ検知

セキュリティやリスク管理、不正監視などの様々な分野において、アノマリ検知は重要である [8]。連続値データに対しては多くの手法が提案されているが [5, 11, 14]、離散値データに対する手法は比較的少ない。Das ら [9] は、ルールベースのアノマリ検知 [15] に基づき、ノイズの多い離散値データからのアノマリ検知手法を提案した。また、CompreX[2] は、MDL [12] を用いたデータ圧縮に基づく、高精度かつ説明力の高いアノマリ検知手法 [13] を発展させた手法である。しかし、いずれも、稀な値・パタンを含む単一のデータの抽出は可能だが、データ集団としてのアノマリを検知することはできない。

2.2 離散値データのクラスタリング

離散値データのクラスタリング手法は多数提案されている [1]。COOLCAT [4] や LIMBO [3] は、我々の提案手法と似た目的関数の最小化を利用する。いずれも、クラスタ数 k に比例した計算量を必要とするため、数百万件以上のデータを含む離散値データから、数千個のデータ集団を、現実的な計算量で生成することはできない。DHCC [16] は階層的にクラスタを生成するアルゴリズムであり、最も良いケースで $log_2 k$ に比例した計算量で計算可能である。しかし、アノマリ検知のために、小規模なデータ集団を補正することは考慮に入っていない。

上記の手法は k を指定する必要があるが、k が数千以上になる場合、k を予め知ることは困難である。MDL に基づき k を自動的に決定する手法が、グラフ分割の文脈からいくつか提案されている [6, 7]。また、ROCAT [10] は、MDL に基づく評価指標を用いる部分空間クラスタリングの手法である。しかし、いずれも、k がデータ数より遥かに少ないことを前提とする。

3 準備

3.1 記法

データセット D に含まれるデータ数を N とする。個々のデータ $r \in D$ は m 個の離散値変数 $\mathcal{F} = \{f_1, \ldots, f_m\}$ を持ち、各変数 $f \in \mathcal{F}$ が取り得る値の集合 $\{v_1, v_2, \ldots\}$ をドメイン $dom(f)$ とする。ドメインは変数間で背反、すなわち $dom(f_i) \cap dom(f_j) = \emptyset, \forall i \neq j$ とする。

クラスタリング C は、互いに共通要素がないデータ集団群 $C = \{C_1, \ldots, C_k\}$ により、データセット D を分割する。すなわち、$C_1 \cup \ldots \cup C_k = D$ であり、$C_p \cap C_q = \emptyset, \forall p \neq q$ である。データ集団 C における、値 v を持つデータ数を、$n(v|C)$ とする。また、データ集団 C に含まれる第 i 番目の変数の値の集合を、$f_i(C)$ とする。

本論文では、全ての対数の底を 2 とし、$0 \log 0 = 0$ とする。

3.2 最小記述長原理

最小記述長原理（MDL） [12] に基づき、与えられたモデルの集合 $\mathcal{M}$ に対し、最良のモデル $M \in \mathcal{M}$ を選択することができる。すなわち、$L(M)$ をモデル M の記述長、$L(D|M)$ をモデル M を用いたときのデータセット D の記述長としたとき、トータルの記述長

$$L(M) + L(D|M),$$

が最小となるモデル M を最良のモデルとして選択する。言い換えれば、MDL は、データセットとモデルそのものの両方を同時に簡潔に表現できるモデルを選択する。データ圧縮に基づくアノマリ検知手法 [13, 2] は、MDL に基づき、データセット D を最も簡潔に表現できるモデル M、すなわちコード表を生成した上で、相対的に記述長の大きいデータをアノマリとして検知する。本研究ではこの手法を拡張し、データ集団ごとに異なるコード表を用いた新たなアノマリ検知手法を提案する。

表 1: コード表 CT の例

データ集団		f_1		f_2		f_3	
C	$code(C)$	v	$code(v\|C)$	v	$code(v\|C)$	v	$code(v\|C)$
C_1	101	a_1	0	b_1	0	c_1	1
				b_2	1	c_2	10
						c_3	11
C_2	10	a_1	0	b_1	0	c_1	1
		a_2	1			c_3	10

4 提案手法

本章では、コード表とデータの記述方法、およびアノマリスコアについて説明する。その後、新規のクラスタリングアルゴリズム、およびデータ集団の可視化方法について述べる。

4.1 コード表とデータの記述方法

まず、コード表を定義する。その後、コード表に基づくデータの記述方法、および MDL に基づくクラスタリングに必要なコード表そのものの記述方法について説明する。

4.1.1 コード表

表 1 は、コード表の例である。コード表 CT は、データ集団に対応した k 個の表のセットであり、それぞれの表は $m+1$ 個に区分されている。最初の区分にはデータ集団そのもののコードが記述され、残りの区分にはデータ集団内の値のコードが記述される。各区分は 2 列の表であり、1 列目にはデータ集団 C または値 v が記述され、2 列目にはデータ集団のコード $code(C)$ または値のコード $code(v|C)$ が記述される。なお、従来手法 [13, 2] と違い、変数間の値の組み合わせに対するコードは用いない。これは、第 4.3 章で説明するクラスタリング手法において、変数間の値の組み合わせを考慮したデータ集団の補正が困難なためである。

4.1.2 データの記述長

コード表 CT を使うことにより、各データ集団 C 中のデータ $r = (v_1, \ldots, v_m)$ を、$code(C)$ に $code(v_1|C), \ldots, code(v_m|C)$ を加えたものとして記述できる。本研究では、$code(v|C)$ を optimal prefix code [12] により記述する。このとき、$code(v|C)$ の記述長は、Shannon エントロピーを用いて

$$L(code(v|C)|CT) = -\log \frac{n(v|C)}{|C|}$$

で算出される。直観的には、データ集団中で高頻度で出現する値のコード長を短くすることにより、データ全体をより圧縮できる。同様に、$code(C)$ の記述長は、

$$L(code(C)|CT) = -\log \frac{|C|}{N}$$

で算出される。直観的には、データセット中で多くのデータを占めるデータ集団のコード長を短くすることにより、データ全体をより圧縮できる。

以上より、データ集団 C 中のデータ $r = (v_1, \ldots, v_m)$ の記述長は、C の記述長と r の記述長の合計

$$L(r|CT) = L(code(C)|CT) + \sum_{i=1}^{m} L(code(v_i|C)|CT)$$

となる。さらに、データセット全体の記述長は、

$$L(D|CT) = \sum_{C \in \mathcal{C}} \sum_{r \in C} L(r|CT)$$

となる。

4.1.3 コード表の記述長

次に、コード表 CT そのものの記述長について説明する。コード表の各区分の 2 列目の記述長は、単純に $code(C)$ と $code(v|C)$ の合計である。コード表の各区分の 1 列目、値 v の記述には、再び optimal prefix code を用いる。すなわち、各値の記述長を、Shannon エントロピーを用いて、CT における各値の出現頻度に基づき算出する。具体的には、コード表 CT における第 i 変数の値 v の記述長は、CT における v の出現数 r_v、および出現数の合計 $c_i = \sum_{v_i \in dom(f_i)} r_{v_i}$ を用いて、$-\log \frac{r_v}{c_i}$ により算出する。以上より、コード表 CT の記述長は、

$$\begin{aligned} L(CT) &= \sum_{C \in \mathcal{C}} L(code(C)|CT) \\ &\quad + \sum_{C \in \mathcal{C}} \sum_{i=1}^{m} \sum_{v \in f_i(C)} L(code(v|C)|CT) \\ &\quad + \sum_{i=1}^{m} \sum_{v \in dom(f_i)} -r_v \log \frac{r_v}{c_i} \end{aligned}$$

となる。

なお、CT の第 1 区分の 1 列目にあたるデータ集団 C の番号は、記述する必要がない。これは、情報の損失なくデータセットを圧縮するためには、C の番号を用いる必要がないためである。すなわち、データが $(code(C), code(v_1|C), \ldots, code(v_m|C))$ で記述されれば、コード表 CT において C が特定でき、さらに $v_1, \ldots, v_m$ を特定することができる。

4.1.4 記述長の合計

データセット D とコード表 CT の記述長の合計は、

$$\begin{aligned} L(CT, D) &= L(CT) + L(D|CT) \\ &= \sum_{C \in \mathcal{C}} -\log \frac{|C|}{N} + \sum_{C \in \mathcal{C}} \sum_{i=1}^{m} \sum_{v \in f_i(C)} -\log \frac{n(v|C)}{|C|} \\ &\quad + \sum_{i=1}^{m} \sum_{v \in dom(f_i)} -r_v \log \frac{r_v}{c_i} \\ &\quad + \sum_{C \in \mathcal{C}} -|C| \log \frac{|C|}{N} \\ &\quad + \sum_{C \in \mathcal{C}} \sum_{i=1}^{m} \sum_{v \in f_i(C)} -n(v|C) \log \frac{n(v|C)}{|C|} \end{aligned} \quad (1)$$

となる。後述するクラスタリング手法は、MDL に基づき、$L(M, D)$ すなわち $L(CT, D)$ を最小とする CT を選択する。

4.2 アノマリスコア

データ集団に与える、2 種類のアノマリスコアを提案する。いずれも、スコアが閾値より大きい集団をアノマリとみなす。

4.2.1 圧縮アノマリスコア

圧縮アノマリスコアは、従来のデータ圧縮に基づくアノマリスコアを、データ集団の記述長を用いて拡張したスコアである。圧縮アノマリスコア $SC_{\text{comp}}(C)$ を、データを集団化しない場合のコード表 CT_1 を用いて、

$$SC_{\text{comp}}(C) = \frac{1}{|C|} \sum_{r \in C} L(r|CT_1) - \frac{1}{|C|} \sum_{r \in C} L(r|CT)$$

と定義する。第 1 項は、従来のデータ圧縮に基づくアノマリスコアに相当するものであり、稀な値を持つデータ集団ほど高い値となる。ただし、従来手法 [13, 2] のように、変数間の値の組み合わせは考慮していない。第 2 項は、集団内のデータ間で値が多く共有されている場合に、大きい値となる。

4.2.2 エントロピーアノマリスコア

圧縮アノマリスコアは、各変数の値の共有のされ方の違いが考慮できない。例えば、送信元 IP が多く共有されているデータ集団が多くを占める場合に、ポート番号が多く共有されているデータ集団をアノマリとして検知することができない。そこで、データ集団 C の第 i 変数の記述長の平均値（エントロピー）

$$H(f_i(C)) = \sum_{v \in f_i(C)} -\frac{n(v|C)}{|C|} \log \frac{n(v|C)}{|C|}$$

に対し、$(H(f_1(C)), \ldots, H(f_m(C)))$ が外れ値となるデータ集団を検知することを考える。このとき、値の共有のされ方はデータ集団のサイズにも依存するため、$L(C|CT)$ も重要な特徴量である。これらのことから、データ集団 C の値の共有のされ方の特徴を、ベクトル $\mathbf{x}_C = (L(C|CT), H(f_1(C)), \ldots, H(f_m(C)))^T$ により表現する。そして、データ集団 C のエントロピーアノマリスコア $SC_{\text{ent}}(C)$ を、マハラノビス距離

$$SC_{\text{ent}}(C) = \sqrt{(\mathbf{x}_C - \mu)^T S^{-1} (\mathbf{x}_C - \mu)}$$

により定義する。このとき、$\mu = \frac{1}{k} \sum_{C' \in \mathcal{C}} \mathbf{x}_{C'}$ であり、S はベクトルの集合 $\mathbf{x}_{C_1}, \ldots, \mathbf{x}_{C_k}$ に関する分散共分散行列である。

4.3 クラスタリングアルゴリズム

我々は、MDL に基づき、$L(CT, D)$ (式 1) を最小にするクラスタリングを探索したい。しかし、全ての可能性のある CT に対して $L(CT, D)$ を算出することは現実的ではなく、また全探索のための有効な枝狩り手法を考えることも困難である。そのため、ヒューリスティクスによる手法 DescCat を提案する。DescCat は、$L(CT, D)$ を小さくするように、データ集団の分割と統合を繰り返す。

4.3.1 メインアルゴリズム

アルゴリズム 1 は、DescCat の疑似コードである。データが集団化されていない状態から開始し、$L(CT, D)$ が減少しなくなるまで、データ集団の分割と統合を繰り返す。

データ集団を統合する最も単純な方法は、全てのデータ集団の組み合わせに対し、統合したときの $L(CT, D)$ を算出することである。しかし、集団数 k に対し少なくとも k^2 に比例する計算量となるため、集団数が数千個以上になる場合には現実的ではない。そこで、DescCat は、データセット全体を、データ集団を単位として分割する。これは、データ集団の分割と同じアルゴリズムで実現できる（アルゴリズム 1 の split()）。その結果生成された「データ集団の集団」は、データ集団を統合したものとみなせる。そして、データ集団の分割前に比べて $L(CT, D)$ が減少しないのであれば、それ以上統合は不可能と判断し、終了する。

Algorithm 1: DescCat

Input: $D = \{r_1, \ldots, r_N\}, \epsilon_1, \epsilon_2, maxiter$
ただし $r_s = (v_{s1}, \ldots, v_{sm}), v_{si} \in dom(f_i)$
Output: $C = \{C_1, \ldots, C_k\}$, コード表 CT, 記述長 L

```
(C, CT, L) ← initialize(D) ; ▷C を 1 つの集団に初期化
for iter ← 1 to maxiter do
    C_0 ← C, CT_0 ← CT, L_0 ← L;
    ▷データ集団の分割
    G ← {{r_1}, ..., {r_N}} ;                ▷データがユニット
    (C, CT, L) ← split (D, C, CT, L, G, ε_1);
    ▷データ集団の統合
    G ← C ;                                  ▷データ集団がユニット
    (C, CT, L) ← initialize(D) ;  ▷C を 1 つの集団に初期化
    (C, CT, L) ←split (D, C, CT, L, G, ε_2);
    if L ≥ L_0 then
        C ← C_0, CT ← CT_0, L ← L_0;
        break;
    end
end
return C, CT, L;
```

4.3.2 split アルゴリズム

アルゴリズム 2 は、分割と統合の両方に用いる split アルゴリズムの疑似コードである。このアルゴリズムは、データ集団のランダム分割とその補正を繰り返す。データは、「ユニット」と呼ぶ単位によりデータ集団に割り当てる。ユニットは、集団の分割では 1 つのデータであり、集団の統合では 1 つのデータ集団である。

補正のアルゴリズムは、基本的には、$L(CT, D)$ が減少しなくなるまで、個々のユニットの交換を繰り返すものである（本論文では詳細は省略する）。しかし、データ集団に数万以上のデータが含まれる場合、$L(CT, D)$ を減少させられるよう補正することは容易ではない。そのため、将来の分割における最終的な $L(CT, D)$ の減少の期待値を導入する。すなわち、(1) 集団が大きいほど、また、(2) ランダム分割に比べて $L(CT, D)$ が大きく減少するほど、将来的にも分割しやすいと仮定する。この仮定に基づき、$L(CT, D)$ の減少の期待値を $\epsilon|C|(L^r - L^u)$ により算出する。ここで、L^r はランダム分割後の $L(CT, D)$、L^u は補正後の $L(CT, D)$ である。そして、$L^u - \epsilon|C|(L^r - L^u)$ が分割前の $L(CT, D)$ より小さいのなら、将来的に良い分割となる可能性が高いとみなし、分割を実行する。なお、ϵ は、データ集団の分割と統合で異なる値でもよい。今回の実験では、共に $\epsilon = 0.01$ を用いて、充分に良い結果を得た。最適な ϵ はデータにより異なり、その決め方は今後の課題である。

4.3.3 計算量

全体の記述長 (式 1) を全て計算しなおす場合、計算量は、$O(N + \sum_{i=1}^{m} |dom(f_i)|)$ である。しかし、1 つのデータが別のクラスタに再配置されるごとに、記述長は $O(1)$ で更新することができる。従って、DescCat の全体の計算量は、I をアルゴリズム 1 の繰り返し回数（上限は $maxiter$）、S をアルゴリズム 1 の 1 回の繰り返しの中で各データが再配置を受ける平均回数とすると、$O(INS)$ となる。最悪は、アルゴリズム 2 の毎回の分割において、生じるデータ集団の一方がそれ以上分割されない場合である。この場合、S は最終的な集団数 k となり、全体の計算量は $O(INk)$ になる。しかし、多くの場合には毎回バランスよく分割されるために S は $\log_2 k$ に近くなり、全体の計算量は $O(IN \log_2 k)$ となる。これは、データ集団数が数千個以上の場合でも、現実的な計算量である。なお、階層的な分割アルゴリズムである DHCC [16] も類似の計算量であるが、DHCC は分割後に統合するプロセスを持たない。

Algorithm 2: split

Input: D, C, コード表 CT, 記述長 $L, \mathcal{G}, \epsilon$
Output: C, コード表 CT, 記述長 L

```
Q ← C ;                                          ▷初期化
while Q ≠ ∅ do
    Q から集団 C_p を選択;
    Q ← Q\{C_p};
    (C^0, C^1, CT^r, L^r) ← ランダム分割 (C_p, CT, G);
    (C^0, C^1, CT^u, L^u) ← 補正 (C^0, C^1, CT^r, G);
    if L^r − ε|C_p|(L^r − L^u) < L then
        C ← C\{C_p} ∪ {C^0, C^1}, CT ← CT^u, L ← L^u;
        Q ← Q ∪ {C^0, C^1};
    end
end
return C, CT, L;
```

表 2: DARPA 侵入検知ログの攻撃のタイプ分類。

タイプ	攻撃
DOS	back, land, neptune, smurf, teardrop, syslog
R2L	dict, dict_simple, ftp-write, guest, imap, phf, multihop, spy, warez, warezclient, warezmaster
U2R	eject, eject-fail, ffb, ffb_clear, format, format_clear, format-fail, loadmodule, load_clear, perl_clear, perlmagic, rootkit
probing	ipsweep, nmap, satan, portsweep
anomaly	anomaly

5 評価実験

アノマリ検知の精度評価結果、および DescCat の計算量と圧縮効率の評価結果を報告する。提案手法は、Python で実装した。全ての実験は、2.8GHz の 6 コアの CPU と 48GB のメモリを持つ、64 ビット Windows 7 の計算機を用いて行った。

5.1 データセット

性質の異なる 2 つの侵入検知ログを用いて実験を行った。

5.1.1 DARPA 侵入検知ログ

DARPA Intrusion Detection Data Sets[1]は、意図的に行われた攻撃が含まれた侵入検知ログであり、侵入検知の精度評価が可

[1]http://www.ll.mit.edu/ideval/data/

表 3: DescCat が DARPA 侵入検知ログで生成したデータ集団の概要。

Week	$\|D\|$	sIP	dIP	sPort	dPort	DOS	R2L	U2R	probing	anomaly	$\|C\|$	$min\|C\|$	$max\|C\|$	$ave\|C\|$
1	177,487	40	1,040	12,795	9	1	1	4	0	0	988	2	104,239	179.6
2	187,986	40	2,265	15,819	108	1,002	55	0	1,043	0	1,410	2	106,970	133.3
3	360,225	1,096	4,987	35,379	7,098	76,638	40	1	7,236	0	2,094	2	159,742	172.0
4	116,881	93	2,133	17,307	11,444	201	1,770	31	11,793	0	2,073	2	25,118	56.4
5	592,611	64	2,022	45,501	8,485	1,174	0	23	7,519	0	2,162	2	201,323	274.1
6	939,247	67	1,978	53,143	12,740	717	883	14	14,166	9	1,135	3	199,325	827.5
7	431,578	54	2,265	36,603	10,141	390	2	4	9,341	0	1,001	2	200,794	431.1

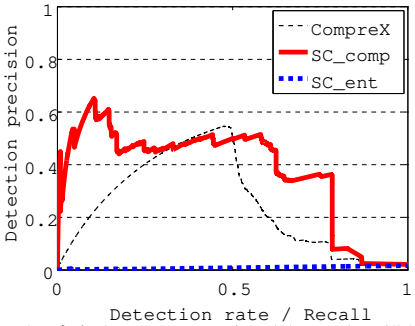


図 2: 再現率-適合率プロットの例（第 6 週）。横軸に再現率、縦軸に適合率を示す。黒色の点線は CompreX、赤色の太い実線は SC_{comp}、青色の太い点線は SC_{ent} である。

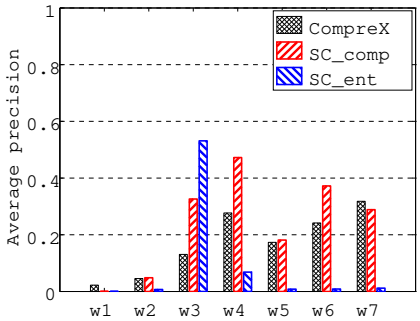


図 3: DARPA 侵入検知ログにおける各手法の平均適合率。横軸に週、縦軸に平均適合率を示す。

表 4: DARPA 侵入検知ログにおける平均精度。

Method	All	DOS	R2L	U2R	Probing	Anomaly
CompreX	0.17	0.04	0.02	**0.05**	**0.19**	0.00
SC_{comp}	**0.24**	**0.09**	**0.08**	0.01	**0.18**	0.00
SC_{ent}	0.09	0.07	0.00	0.00	0.02	0.00

能である。今回は、1998 年の 7 週間分の学習データを用いた。なお、著名な KDD Cup 99 データセットは、このデータセットを専門家が人手で加工したものである。我々は、専門家の知識に頼らないアノマリ検知を評価するため、送信元 IP（sIP）、送信先 IP（dIP）、送信元ポート番号（sPort）、送信先ポート番号 (dPort) の 4 変数を加工せずに用いた。また、Attack Database[1]を参考に、攻撃を 5 つに分類した（表 2）。ただし、本来 *DOS* 攻撃に分類される *neptune* は、正常なデータとした。なぜなら、*neptune* 攻撃が発生するときは、明らかにそれとわかるデータが、全てのデータの半分以上を埋め尽くすからである。表 3 に、各攻撃タイプのデータ数を示す。

5.1.2 クラウド侵入検知ログ

我々は、ある企業が提供するクラウド環境の侵入検知ログへの適用実験も行った。このデータセットは、送信元 IP（sIP）、送信先 IP（dIP）、送信先ポート番号（dPort）、シグネチャ（Sig）の、4 つの変数を持つ。シグネチャには、侵入検知システムが推定した攻撃タイプが示されている。約 2,000,000 件のデータが含まれる、6 週間分のデータを実験に用いた。クラウド侵入検知ログの概要を、表 5 に示す。

5.2 DARPA 侵入検知ログからのアノマリ検知

表 3 に、DescCat により生成されたデータ集団数、およびデータ集団のデータ数の最小値、最大値、平均値を示す。データ集団数は、どの週でもおよそ 1,000 個から 2,000 個であった。比較手法として、CompreX [2] のプログラムをダウンロードして用いた。なお、CompreX を大規模データに適用するため、主要アルゴリズムには影響しない微小な改変を加えた。

5.2.1 評価指標

精度評価の指標として、[2] と同様の平均適合率を用いた。これは、再現率に対する適合率の平均値のことである。適合率とは、アノマリとしたデータのうち、正しく攻撃であったデータが占める割合である。また、再現率とは、全攻撃のうち、検知できた攻撃が占める割合である。具体的には、まず、アノマリスコアの様々な閾値に対する再現率と適合率を算出する。そして、再現率を横軸に、適合率を縦軸としたプロットを作成し、プロットの下部の面積を、平均適合率とする。図 2 は、再現率-適合率プロットの例である（第 6 週目）。下部の面積が大きいほど、手法として性能が良いことを意味する。この場合、SC_{comp} はほとんどの再現率に対して従来手法より高い適合率を示す一方、SC_{ent} は従来手法の適合率より低い。

5.2.2 精度評価

表 4 には、平均適合率の全ての週の平均値を示している (“*All*”)。全体として、SC_{comp} は従来手法より平均適合率が高く、SC_{ent} は従来手法より低い。攻撃タイプ別に見ると、SC_{comp} は、*DOS* タイプと *R2L* タイプを、従来手法より良い精度で検知している（表 4）。以下、SC_{comp} と SC_{ent} の違いについて、詳細に分析する。

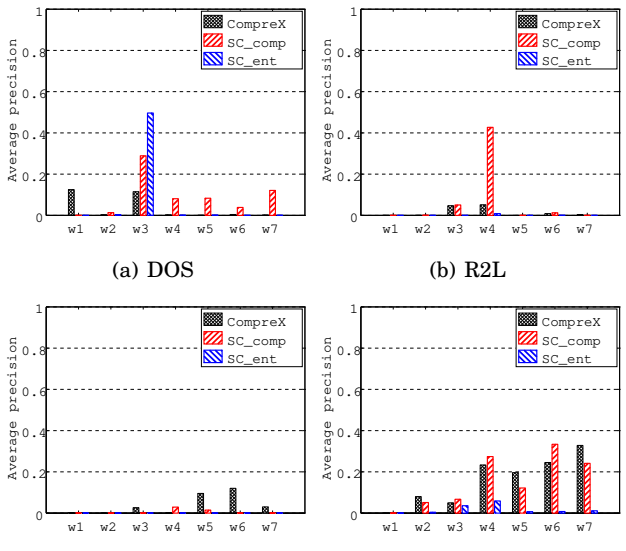
図 3 に、週別の平均適合率を示す。第 3 週と第 4 週、および第 6 週において、SC_{comp} は従来手法より良い結果を示している。また、SC_{ent} は、第 3 週においては、どの手法よりも高い平均適合率を示している。図 4 は、週別の平均適合率を、さらに攻撃タイプ別に示したものである。ただし、*anomaly* タイプの攻撃は、どの手法でも平均適合率が著しく低いため、示していない。第 3 週における SC_{ent} の高い平均適合率は、*DOS* タイプの検知によるもの

表 5: クラウド侵入検知ログから DescCat により生成されたデータ集団の概要。

$\|D\|$	sIP	dIP	dPort	Sig	$\|C\|$	$min\|C\|$	$max\|C\|$	$ave\|C\|$
1,942,041	123,156	3,935	50,217	3,640	13,565	2	114,952	143.2

表 6: クラウド侵入検知ログの、SC_{comp} 上位 5 個のデータ集団。

SC_{comp}	$\|C\|$	sIP	dIP	dPort	Sig	説明
34.9	4	*.*.46.146, *.*.182.175, *.*.2.189	*.*.241.196, *.*.248.150	764, 847	[TCP***]	RST パケットの大量送信
31.9	4	*.*.65.129	*.*.240.19, *.*.240.15	5900	[VNC***]	VNC(Virtual Network Computing) に対する攻撃
31.0	4	*.*.200.65, *.*.97.66	*.*.248.165	1198, 1226, 1164, 1249	[Internet***], [Malware***]	悪意あるスクリプト挿入とマルウェアのダウンロード
30.2	2	*.*.0.0	*.*.0.0	0	[Invalid***]	不正な IP バージョン番号
29.7	6	*.*.227.13, *.*.9.196, *.*.153.106, *.*.57.142, *.*.227.13, *.*.18.234	*.*.247.211	10649, 2808, 33909, 31389, 37446, 5712	[Malicious***]	悪意ある JavaScript 挿入



(a) DOS (b) R2L

(c) U2R (d) Probing

図 4: DARPA 侵入検知ログの、攻撃タイプ別の平均適合率。横軸に週、縦軸に平均適合率を示す。(a) は *DOS* タイプ、(b) は *R2L* タイプ、(c) は *U2R* タイプ、(d) は *probing* タイプである。

であった（図 4(a)）。具体的には、SC_{ent} は、*smurf* 攻撃をほぼ正確に検知していた。すなわち、75,630 件の全ての *smurf* 攻撃は、併せて 75,636 個のデータを含む SC_{ent} が高い上位 2 つのデータ集団に含まれていた。その他、SC_{comp} は、第 4 週において、*R2L* タイプの攻撃を特に精度良く検知していた（図 4(b)）。この週で最も多い *R2L* タイプの攻撃は *warezclient* 攻撃であり（1,766 件）、これらのほぼ半数が、SC_{comp} の上位 100 データ集団（1,474 件）に含まれていた。このように、SC_{comp} と SC_{ent} は、それぞれ異なるタイプの攻撃パタンを的確に捉えられることが示唆された。

さらに、SC_{comp} と SC_{ent} の上位のデータ集団において、各攻撃がどのように含まれているかを、具体的に示す。図 5(a) は、第 6 週における、SC_{comp} の上位 20 データ集団である。多くのデータ集団は *probing* タイプの攻撃を含み、それらは全て *satan* 攻撃であった。第 18 位のデータ集団（図 5(a) 矢印）は *DOS* タイプの *land* 攻撃であり、第 6 週の 17 個の *land* 攻撃のうち 6 個を含んでいた。また、同じ週における、SC_{ent} の上位 20 個のデータ集団を図 5(b)（(c) は拡大図）に示す。第 15,16,17 位のデータ集団の全てのデータは *DOS* 攻撃であった（図 5(b)(c) 矢印）。これらは *teardrop* 攻撃であり、600 個の全ての *teardrop* 攻撃が含まれていた。以上の結果は、各データ集団により、攻撃に関連するデータをうまく切り出せていることを示唆している。

5.3 クラウド侵入検知ログからのアノマリ検知

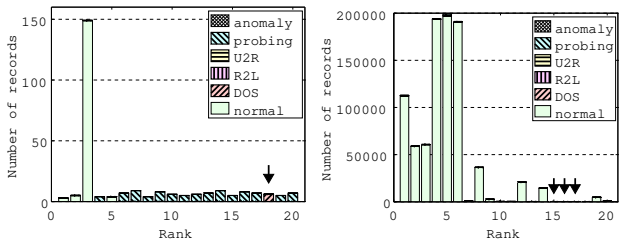
続いて、クラウド侵入検知ログの実験結果を報告する。約 2,000,000 個のデータが、10,000 個以上のデータ集団に分割された。表 5 は、DescCat によるクラスタリングの結果の概要である。クラウド侵入検知ログについては、定量的な評価のための正解データが無いため、定性的な結果のみ示す。

5.3.1 圧縮アノマリスコア

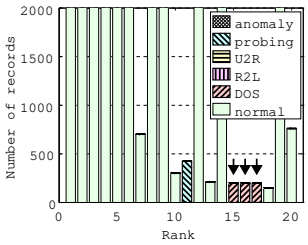
表 6 に、SC_{comp} の上位 5 個のデータ集団の概要を示す。このクラウド環境の侵入検知システムをよく知る専門家による、各データ集団の解釈も示す。例えば、SC_{comp} が最も高いデータ集団には、3 種類の送信元 IP と 2 種類の送信先 IP と 2 種類の送信先ポート番号、および 1 種類のシグネチャを持つ 4 つのデータが含まれる。いずれも、データセット中で稀な値を含み、かつ互いに値を共有しあうデータ集団であるが、大量の稀な値を含むデータセットの中から、このような小規模のデータ集団に注目し分析することは容易ではない。

5.3.2 エントロピーアノマリスコア

SC_{ent} が高いデータ集団を専門家が分析した結果、いくつかの注目すべき攻撃パタンが発見された。その 1 つは、可能性のある ID/PASS の組合せを手当たり次第に試行する総当たり攻撃であった（図 6(a)）。すなわち、2 つの攻撃元（“*.*.177.228” と “*.*.12.2”）が、2 つの標的（“*.*.248.88” と “*.*.252.222”）に対し、総当たり攻撃を行っていた。さらに、そのうち 1 つの攻撃元は、別の標的（“*.*.247.239”）に対しても類似の攻撃を行っていた（図 6(a) の赤色のノード）。なお、これは、SC_{ent} の上位約 1.0% にあたる、第 140 位のデータ集団であった。



(a) SC_{comp} (b) SC_{ent}



(c) SC_{ent} (zoom)

図 5: 第 6 週におけるアノマリスコアの上位 20 個のデータ集団。(a) は、SC_{comp} の上位データ集団である。1 つのデータ集団には、*DOS* タイプの攻撃が多く含まれていた（矢印）。(b) は、SC_{ent} の上位データ集団である（(c) は拡大図）。いくつかのデータ集団には、(a) とは別の *DOS* タイプの攻撃が多く含まれていた（矢印）。

また、稀なタイプの Secure Shell(SSH) 総当たり攻撃も発見された（図 6(b)）。すなわち、ある 1 組の標的群に対し、多数の使い捨て IP アドレスを用いて、可能性のある SSH プロトコルのパスワードを試行していた（図 6(b) の “[Secure***]” に接続するデータ）。さらに、図 6(b) の赤色のノードにより、この攻撃にいくつかのホスト探索が関与していることに、分析者は容易に気づくことができた。さらなる分析により、このホスト探索は、この攻撃の前触れとして行われていることが明らかになった。なお、これは、SC_{ent} の上位約 0.9% にあたる、第 118 位のデータ集団であった。

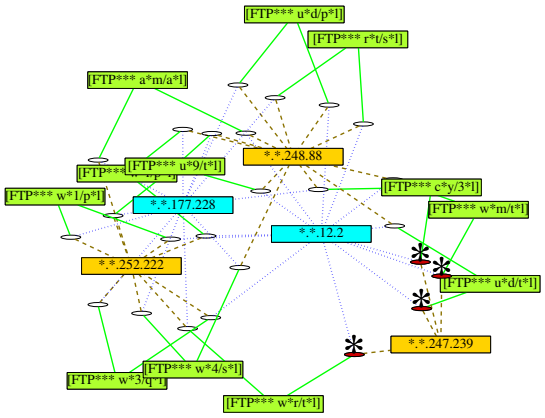
5.4 計算量と圧縮効率の評価実験

DARPA 侵入検知ログを 1 日単位で分割し、計算量と圧縮効果を評価した。1 日あたり 1,000 個から 500,000 個のデータが含まれ、DescCat により 500 個から 1,300 個のデータ集団が生成された。

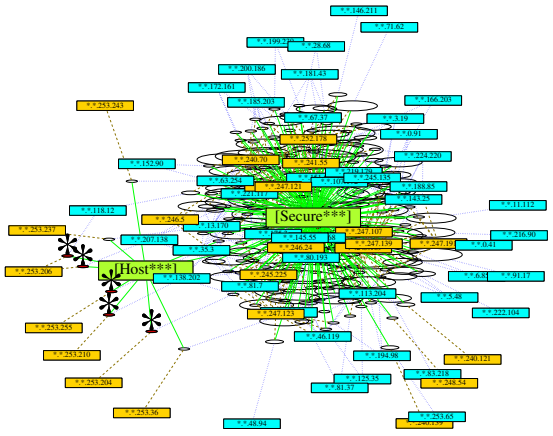
比較対象として、類似のコスト関数を最小化する COOLCAT [4] を用いた。ただし、コスト関数は $L(D, CT)$ とした。以降、この比較手法を DescCat-CC とよぶ。DescCat-CC は、データを 1 つずつ含む初期データ集団を生成した後、残りのデータを追加していく。全てのデータの追加後、各データ集団で稀な値を持つデータを、適切なデータ集団に再配置する。DescCat-CC のパラメータは、データ集団の最小サイズ m を 100、確信度 δ を 0.5、再配置するデータの割合を 10% とした（詳細は [4] 参照）。また、DescCat-CC は予めデータ集団数 k を与える必要があるが、今回は DescCat で生成されたデータ集団数と同じ数を与えた。今回の実験では、*maxiter* を 2 とすることで、充分な性能が得られた。

5.4.1 計算量の評価

図 7(a) に、DescCat と DescCat-CC の計算時間を示す。共に、計算時間は全データ数にほぼ比例して増加するが、DescCat は DescCat-CC に比べて傾きが小さい。これは、DescCat-CC の計算量はデータ集団数 k に比例するのに対し、DescCat の計算量は最善の場合に $\log_2 k$ に比例するためと考えられる。



(a) 攻撃パタン 1



(b) 攻撃パタン 2

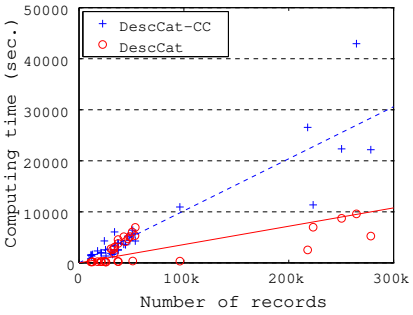
図 6: クラウド侵入検知ログで発見されたデータ集団。全ての変数の値が一致するデータを 1 つの楕円のノードでまとめて示し、ノードの大きさはデータ数を表す。値は長方形のノードで示し、変数により色分けしている。データと、そのデータに含まれる値をエッジで接続する。集団内での値 v の情報量の合計 $-n(v|C)\log\frac{n(v|C)}{|C|}$ が閾値より小さい場合には、v を省略している。情報量の多いデータは赤いノードで示し、さらにアスタリスクで強調する。

5.4.2 圧縮効率の評価

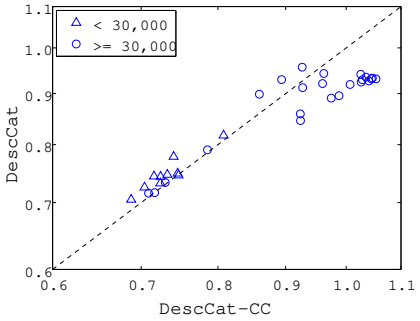
図 7(b) は、DescCat と DescCat-CC の圧縮率の比較である。ここで、圧縮率とは、クラスタリング前後の記述長の比である。圧縮率が小さいほど、良く圧縮されたことを意味する。DescCat は、データ数が 30,000 個より大きいデータセットに対し、DescCat-CC より良く圧縮できた。より良く圧縮できているほど、提案手法が意図するアノマリを精度良く検知できる可能性が高い。総じて、DescCat は、大規模なデータセットでは、計算量と圧縮効率の両面で従来手法よりも良い性能を示すことがわかった。

6 結論

データ集団別のコード表を用いた、新たなアノマリ検知手法を提案した。提案手法は、大規模な侵入検知ログに対し、従来手法より高い検知精度を示した。また、大量の小規模なデータ集団を効率的に生成する DescCat を提案した。DescCat は、大規模データに対し、計算量と圧縮効率の両面で、従来手法より良い性能を示した。今回の提案手法は変数間の値の組み合わせを考慮に入れていないために、従来手法で検知できていたアノマリを見逃して



(a) Computing time



(b) Compression ratio

図 7: (a) 1 日単位の DARPA 侵入検知ログに対する計算時間。DescCat の回帰直線は赤色の実線、DescCat-CC の回帰直線は青色の点線で示す。横軸にデータセットのサイズ、縦軸に計算時間を示す。(b) クラスタリング前の記述長に対する、クラスタリング後の記述長の比（圧縮率）。データ数が 30,000 より少ないデータセットは三角、30,000 以上のデータセットは丸で示す。横軸に DescCat-CC の圧縮率、縦軸に DescCat の圧縮率を示す。

いる可能性がある。今後、データ集団別のコード表に、変数間の値の組み合わせを考慮することを検討する。

[文献]

[1] Charu C. Aggarwal and Chandan K. Reddy, editors. *Data Clustering: Algorithms and Applications*. CRC Press, 2014.

[2] Leman Akoglu, Hanghang Tong, Jilles Vreeken, and Christos Faloutsos. Fast and reliable anomaly detection in categorical data. In *CIKM*, pages 415–424, 2012.

[3] Periklis Andritsos, Panayiotis Tsaparas, Renée J. Miller, and Kenneth C. Sevcik. LIMBO: scalable clustering of categorical data. In *EDBT*, pages 123–146, 2004.

[4] Daniel Barbará, Yi Li, and Julia Couto. COOLCAT: an entropy-based algorithm for categorical clustering. In *CIKM*, pages 582–589, 2002.

[5] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. LOF: identifying density-based local outliers. *SIGMOD Record*, 29:93–104, May 2000.

[6] Deepayan Chakrabarti. Autopart: Parameter-free graph partitioning and outlier detection. In *PKDD*, pages 112–124, 2004.

[7] Deepayan Chakrabarti, Spiros Papadimitrou, Dharmendra Modha, and Christos Faloutsos. Fully automatic cross-associations. In *KDD*, pages 79–88, 2004.

[8] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3), 2009.

[9] Kaustav Das, Jeff G. Schneider, and Daniel B. Neill. Anomaly pattern detection in categorical datasets. In *KDD*, pages 169–176, 2008.

[10] Xiao He, Jing Feng, Bettina Konte, Son T. Mai, and Claudia Plant. Relevant overlapping subspace clusters on categorical data. In *KDD*, pages 213–222, 2014.

[11] Spiros Papadimitriou, Hiroyuki Kitagawa, Phillip B. Gibbons, and Christos Faloutsos. Loci: Fast outlier detection using the local correlation integral. In *ICDE*, pages 315–326, 2003.

[12] J. Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465 – 471, 1978.

[13] Koen Smets and Jilles Vreeken. The odd one out: Identifying and characterising anomalies. In *SDM*, pages 804–815, 2011.

[14] Ye Wang, Srinivasan Parthasarathy, and Shirish Tatikonda. Locality sensitive outlier detection: A ranking driven approach. In *ICDE*, pages 410–421, 2011.

[15] Weng-Keen Wong, Andrew W. Moore, Gregory F. Cooper, and Michael M. Wagner. Rule-based anomaly pattern detection for detecting disease outbreaks. In *AAAI/IAAI*, pages 217–223, 2002.

[16] Tengke Xiong, Shengrui Wang, André Mayers, and Ernest Monga. A new mca-based divisive hierarchical algorithm for clustering categorical data. In *ICDM*, pages 1058–1063, 2009.

丸橋 弘治　Koji MARUHASHI
株式会社富士通研究所人工知能研究所所属．1999 年京都大学大学院理学研究科修士課程修了．同年富士通株式会社入社．2002 年より株式会社富士通研究所．2009 年米国カーネギーメロン大学客員研究員．2014 年筑波大学大学院システム情報工学研究科博士後期課程修了．博士（工学）．2016 年より北陸先端科学技術大学院大学客員准教授．現在，データマイニング・機械学習の研究開発に従事．日本データベース学会，情報処理学会，各会員．

齊藤 聡美　Satomi SAITO
富士通研究所セキュリティ研究所所属．2015 年 4 月より横浜国立大学大学院環境情報学府博士課程後期在籍．2012 年横浜国立大学大学院博士課程前期修了．同年株式会社富士通研究所入社．ネットワークセキュリティ分析・検知技術の研究開発に従事．情報処理学会会員．

鳥居 悟　Satoru TORII
株式会社富士通研究所セキュリティ研究所所属．1985 年株式会社富士通研究所に入社．現在，サイバーセキュリティの研究開発に従事．情報処理学会会員．

武仲 正彦　Masahiko TAKENAKA
株式会社富士通研究所セキュリティ研究所所属，現所長．1992 年大阪大学大学院工学研究科博士前期課程修了．2009 年筑波大学システム情報工学研究科博士後期課程修了．博士（工学）．1992 年より富士通株式会社および株式会社富士通研究所に勤務．2014 年より情報セキュリティ大学院大学連携教授．情報セキュリティ，暗号実装，サイバーセキュリティの研究に従事．富士通社内制度セキュリティマイスタで，グローバルホワイトハッカー認定者．2005 年電気科学技術奨励賞受賞．2013 年情報処理学会喜安記念業績賞受賞．電子情報通信学会，情報処理学会，デジタル・フォレンジック研究会，各会員．