埋め込み表現を用いたIoTデバイス異常動作検知手法の改良

江田 琉聖[1,a)]　戸川 望[1,b)]

概要：近年，Internet of Things (IoT) デバイスの普及に伴い，ハードウェアデバイスにおけるセキュリティ上の課題が顕在化している．ハードウェアデバイスの消費電力波形を用いて異常動作を検知する手法として，消費電力波形を低次元の埋め込み表現に変換し，その埋め込み表現の予測誤差に基づいて異常動作を検知する手法が提案されている．本稿では，その手法を発展させ，消費電力波形を埋め込み表現に変換するエンコーダと，予測器の両方に改良を加えた新しい手法を提案する．提案手法を Raspberry Pi4 を用いて実装した IoT デバイスに適用した結果，手法を改良したことで，異常動作検知精度の向上を実現した．

キーワード：異常動作検知，サイドチャネル解析，時系列埋め込み，Transformer

Improved IoT Anomalous Behavior Detection Utilizing Embedded Representations

Ryusei Eda[1,a)]　Nozomu Togawa[1,b)]

Abstract: With the spread of Internet of Things (IoT) devices, security issues in hardware devices have become apparent in recent years. As a method for detecting anomalous behaviors using the power waveform of hardware devices, there has been proposed a method that converts the power consumption waveform into low-dimensional embedded representations and detects anomalous behaviors based on the prediction-errors of those embedded representations. In this paper, we improve both of its encoder and predictor and achieve more accurate anomalous behavior detection. The experimental evaluations are demonstrated by applying the proposed method to the IoT device implemented using a Raspberry Pi4.

Keywords: anomalous behavior detection, side-channel analysis, embedded representations, Transformer

1. はじめに

近年，IoT (Internet of Things) デバイスの普及が急速に進み，家庭や産業，医療などの様々な分野で革新的な応用が展開されている．一方で，IoT デバイスの普及に伴い，セキュリティや信頼性に関する課題が顕在化している．特に，IoT デバイスの不正操作や異常動作は，システム全体の信頼性や安全性に深刻な影響を及ぼす可能性があり，これらの課題に対する解決策が求められている [1, 2]．さらに，OTA (Over-The-Air) アップデートを通じた悪意あるコードの配信リスクも高まっており，製造後も継続的に異常動作を検知・監視する技術が対策として求められている [3]．

[1] 早稲田大学基幹理工学研究科情報理工・情報通信専攻
Dept. Computer Science and Communications Engineering, Waseda University
[a)] ryusei.eda@togawa.cs.waseda.ac.jp
[b)] ntogawa@waseda.jp

IoT デバイスの異常動作を検知する手法として，デバイスが生成するデータを分析する手法がいくつか提案されている [4–6]．中でも電力情報は IoT デバイスから容易に取得でき，低コストで異常動作検知が可能となる．例えば，ハードウェアトロイ [7] やマルウェアが挿入されることで回路内の配線やゲートの電力消費が変化し，その変化が電力情報として現れる．電力情報を測定することで，ハードウェアトロイやマルウェアの存在，あるいはこれらによる異常動作を検知できる可能性がある．以上の点より，本稿では，IoT デバイスの消費電力を用いた異常動作検知手法に注目する．

電力データは時間の経過に伴って測定されるため，典型的な時系列データとして扱える．よって，消費電力を用いた IoT デバイスの異常動作検知問題は，時系列データの異常検知問題と捉えられる．時系列データを用いた異常検知手法はいくつか提案されているが，いずれの手法も時系列

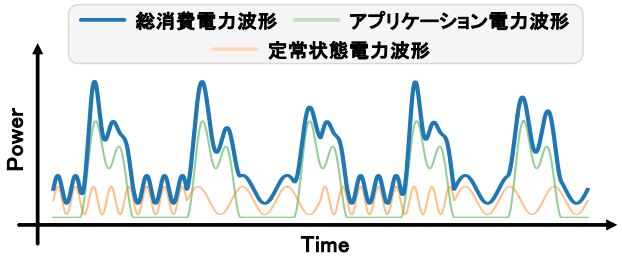


図 1: IoT デバイスの消費電力波形の例.

データの長期的な依存関係から異常を検知できない [8–15].

我々は，消費電力波形を低次元の埋め込み表現に変換し，時系列データの長期的な依存関係を利用して IoT デバイスの異常動作を検知する手法を提案している [16]．本稿では，この手法をさらに発展させ，消費電力波形の埋め込みを行うエンコーダと埋め込み表現予測を行う予測器の両方に改良を施した新手法を提案する．提案手法では，まず，エンコーダに CNN (Convolutional Neural Network) を導入することで，消費電力波形の本質的な特徴を保持しつつ効率的に圧縮された埋め込み表現の獲得を実現する．さらに，予測器に Transformer アーキテクチャの概念を導入することで，埋め込み表現間の複雑な関係性や時系列的な意味的つながりを効果的に捉えた予測を可能とする．実験の結果，提案手法は従来手法 [16] と比較して，より表現力の高い埋め込みを実現し，異常動作検知精度の向上を達成した．

本稿の主な貢献は以下の通りである．

(1) CNN ベースエンコーダと Transformer ベース予測器を組み合わせることで埋め込み表現の質を向上させ，消費電力データの長期的な依存関係から高精度に異常動作を検知する手法を提案した．

(2) 実験の結果，提案手法は従来手法 [16] と比較して，より表現力の高い埋め込みを実現し，異常動作検知精度の向上を達成した．

2. 問題定義と関連研究

2.1 定常状態を持つ IoT デバイスに対する異常動作検知問題

IoT デバイスの消費電力波形の例を図 1 に示す．IoT デバイスの消費電力は次に示す 2 つの成分で構成される:

定常状態電力 P_{steady}: 搭載されている OS やハードウェアにより定常的に消費される電力

アプリケーション電力 P_{app}: アプリケーションが実行されることにより消費される電力

消費電力波形は連続した消費電力値で構成される．時刻 i における定常状態電力を $p_{\text{s},i}$，アプリケーション電力を $p_{\text{a},i}$，総消費電力を $p_{\text{t},i}$ とすると，定常状態電力 P_{steady}，アプリケーション電力 P_{app}，総消費電力 P_{total} は次の通りに表せる:

$$P_{\text{steady}} = \{p_{\text{s},1}, \ldots, p_{\text{s},L-1}, p_{\text{s},L}\} \quad (1)$$

$$P_{\text{app}} = \{p_{\text{a},1}, \ldots, p_{\text{a},L-1}, p_{\text{a},L}\} \quad (2)$$

$$\begin{aligned} P_{\text{total}} &= \{p_{\text{s},1} + p_{\text{a},1}, \ldots, p_{\text{s},L-1} + p_{\text{a},L-1}, p_{\text{s},L} + p_{\text{a},L}\} \\ &= \{p_{\text{t},1}, \ldots, p_{\text{t},L-1}, p_{\text{t},L}\} \end{aligned} \quad (3)$$

ここで，L は消費電力波形の長さとする．

近年の IoT デバイスは OS や LED，ファンなどのハードウェアが搭載されており，これらは常時小さな動作を実行している．OS やハードウェアにより定常的に消費される電力を定常状態電力 (P_{steady}) と呼び，複数の定常状態を持つデバイスも存在する．総消費電力波形は，アプリケーション電力と定常状態電力の 2 つの信号源を含むため，図 1 のような複雑な形状となる．

ハードウェアトロイやマルウェアといった異常動作の影響はアプリケーション電力に現れるため，消費電力を測定し分析することでそれらによる異常動作を検知できる可能性がある．

以上の議論により，IoT デバイスの異常動作検知問題を次のように定義する．

定義 1. IoT デバイスの異常動作検知問題とは，複数のアプリケーションが動作する IoT デバイスが与えられた時に，その消費電力を測定，分析することで異常動作を検知する問題である．

特に出荷後の IoT デバイスに対してリアルタイムに異常動作検知することを目標とする．

2.2 関連研究

時系列データの複雑性の増加に伴い，深層学習を活用した異常検知手法が数多く提案されてきており，これらの手法は，主に予測ベース [8–10, 16]，再構成ベース [11–13]，分類ベース [14, 15] の 3 つに分類できる．

予測ベースの手法は過去の時系列データから将来の値を予測し，実際の観測値との誤差に基づいて異常を検知する [8–10]．再構成ベースの手法では，オートエンコーダなどを用いて入力データを低次元の潜在空間に圧縮し，再び元の次元に復元したときの再構成誤差に基づいて異常を検知する [11–13]．分類ベースの手法では，オートエンコーダなどを用いて入力データから特徴量を抽出し，その分布に基づいて異常を検知する [14, 15].

IoT デバイスには同じ動作を繰り返し実行するという特性があり，その異常動作は単発的な値の逸脱よりも，動作の順序や本来の動作シーケンスからの逸脱として現れる場合が多い [17, 18]．そのため，消費電力を用いた IoT デバイスの異常動作検知において，時系列データの時間的依存関係を捉えるのが重要である．

分類ベースの手法は，[15] のように時系列データを部分波形に分割し，部分波形を独立に扱うため時系列データの時間的依存関係を捉えられず，再構成ベースの手法は高表現力モデルが異常パターンも正確に再構成してしまう問題がある．一方，予測ベースの手法は時系列データの時間的依存関係に基づく異常検知が可能であり，IoT デバイス

の異常動作検知により適している．しかし，より長い時系列データを扱う場合，系列の複雑さが増大し，既存の予測ベースの手法 [8–10] では，予測精度が下がるだけでなく，大規模な学習モデルが必要となり，長期的な時間的依存関係の学習に限界がある．

時系列データの長期的な時間的依存関係を考慮するため，消費電力波形を埋め込み表現に変換し，その埋め込み表現の予測誤差に基づいて異常動作を検知する予測ベースの手法 [16] が提案されているが，消費電力波形を埋め込み表現に変換するエンコーダと，埋め込み表現予測に使用する予測器は単純な機械学習構造になっており，改善の余地がある．上記の議論に基づき，本稿では従来手法 [16] を改良することで，埋め込み表現予測と異常動作検知の精度を向上させた，埋め込み表現を用いた IoT デバイス異常動作検知手法を提案する．

3. 埋め込み表現を用いた異常動作検知手法 [16]

本章では，従来手法となる，埋め込み表現を用いた異常動作検知手法 [16] を説明する．

3.1 Step 0: エンコーダと予測器の準備

従来手法では，エンコーダ (*Encoder*) を用いて消費電力波形を埋め込み表現に変換し，予測器 (*Predictor*) による埋め込み表現の予測誤差に基づいて異常動作を検知する．Step 0 では，異常動作検知対象のデバイスの消費電力を測定し，消費電力波形を埋め込み表現に変換するエンコーダ (*Encoder*) と，埋め込み表現の予測に用いる予測器 (*Predictor*) の準備をする．

まず，異常動作検知対象のデバイスの消費電力を測定する．測定された消費電力データは微小な電力変動やノイズが含まれている．そのため，KZ フィルタ [19] を用いて波形の平滑化を施し，消費電力波形の形状的特徴を捉えやすくする．また，正規化も施し，消費電力データが最小値 0，最大値 1 を持つようにする．平滑化，正規化された消費電力波形を P_{total} とする．P_{total} は式 (3) のように L 個のサンプルから成ることとする．

3.1.1 Step 0.1: エンコーダの準備

エンコーダは消費電力波形を低次元の埋め込み表現に変換する．従来手法では，オートエンコーダを利用して消費電力波形を埋め込み表現に変換する．

まず，測定した長さ L の消費電力波形から長さ W の部分波形をスライディングウィンドウ形式で抽出し，オートエンコーダの学習データを作成する．W は総消費電力波形の支配的な周波数に対する周期とする [20]．入力データと教師データを同一の部分波形にし，損失関数として MSE (Mean Squared Error) を利用することで，オートエンコーダの入力と出力が等しくなるように学習させる．

再構成誤差の最小化を目的として学習されるオートエンコーダは，限られた表現空間内で消費電力波形の本質的な構造情報を優先的に捉えようとする．その結果，定常状態電力といった微小な電力変動など，再構成への寄与が少ない成分は自然と除去され，入力データから抽象的な特徴が抽出される [21]．よって，オートエンコーダによって抽出される特徴量は比較的消費電力の大きいアプリケーションの動作に起因する電力波形の特徴を有する．

3.1.2 Step 0.2: 予測器の準備

続いて，複数の部分波形の埋め込み表現を入力として受け取り，続く埋め込み表現を予測する予測器の準備を行う．

まず，測定した消費電力波形からから長さ W ごとに部分波形を抽出する．1 つの消費電力データからは $\lfloor \frac{L}{W} \rfloor$ 個の部分波形が得られ，抽出された部分波形全体の集合 P_{partial} は次のように表せる．

$$P_{\mathrm{partial}} = \{P_{\mathrm{p},1}, P_{\mathrm{p},2}, ..., P_{\mathrm{p},\lfloor \frac{L}{W} \rfloor - 1}, P_{\mathrm{p},\lfloor \frac{L}{W} \rfloor}\} \tag{4}$$

$$P_{\mathrm{p},i} = \{p_{\mathrm{t},i\times W}, p_{\mathrm{t},i\times W+1}, ..., p_{\mathrm{t},i\times W+W-1}\} \tag{5}$$

その後，抽出した部分波形 $P_{\mathrm{p},i}$ を式 (6) に示すようにそれぞれ Step 0.1 で得たエンコーダを用いて埋め込み表現 $\boldsymbol{z}_i$ に変換する．以降，$\boldsymbol{z}_i$ を i 番目の埋め込み表現と呼ぶこととする．

$$Encoder(P_{\mathrm{p},i}) = \boldsymbol{z}_i \tag{6}$$

i を 1 から $\lfloor \frac{L}{W} \rfloor$ まで 1 ずつ大きくし，埋め込み表現に変換していくことで，埋め込み表現列 $\boldsymbol{Z}$ が得られる．

$$\boldsymbol{Z} = \{\boldsymbol{z}_1, \boldsymbol{z}_2, ..., \boldsymbol{z}_{\lfloor \frac{L}{W} \rfloor - 1}, \boldsymbol{z}_{\lfloor \frac{L}{W} \rfloor}\} \tag{7}$$

続いて，得た埋め込み表現列を用いて予測器の学習を行う．本節では，k 番目，$k+1$ 番目の埋め込み表現 $(\boldsymbol{z}_k, \boldsymbol{z}_{k+1})$ を入力として受け取り，$k+2$ 番目の埋め込み表現 $(\boldsymbol{z}_{k+2})$ を予測するような予測器を想定するが，入力として受け取る埋め込み表現の数や，予測する埋め込み表現の数は柔軟に変更可能であり，適当に設定するものとする．

$\boldsymbol{z}_k, \boldsymbol{z}_{k+1}$ を予測器の入力データ，続く埋め込み表現 $\boldsymbol{z}_{k+2}$ を教師データ，損失関数として MSE を利用し，予測器の出力と，続く埋め込み表現が等しくなるように学習させる．

エンコーダと予測器の準備が完了したら，次に示す Step 1 から Step 4 の手順で異常動作を検知する．

3.2 Step 1: 対象デバイスの消費電力の測定

Step 1 では異常動作検知対象のデバイスの消費電力を測定し，波形の分割を行う．

まず，異常動作検知に使用する消費電力を Step 0 と同様の手順で得る．続いて，式 (4)，式 (5) に示すように，長さ L の消費電力波形を長さ W ごとに分割し，$\lfloor \frac{L}{W} \rfloor$ 個の部分波形を得る．

3.3 Step 2: 消費電力波形の埋め込み

Step 2 では，消費電力波形を埋め込み表現に変換する．

式 (6) に示すように Step 0 で得たエンコーダに Step 1 で得た部分波形を入力し，消費電力波形の部分波形を埋め込み表現に変換する．式 (7) のように，1 つの電力データ

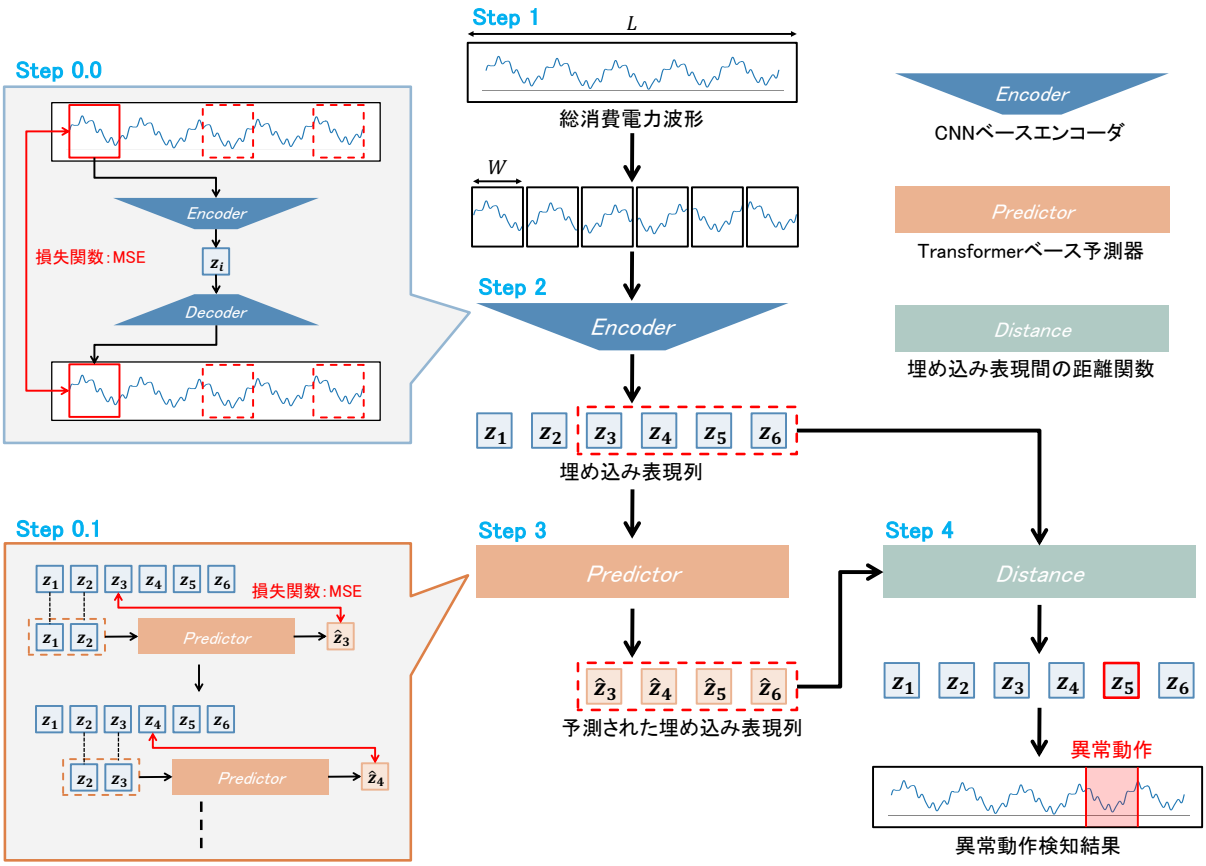


図 2: 提案手法の構成図.

あたり $\lfloor \frac{L}{W} \rfloor$ 個の埋め込み表現から成る 1 つの埋め込み表現列が得られる．1 つの埋め込み表現は m $(1 < m < W)$ 次元のベクトルで構成され，Step 2 で得られた埋め込み表現を $\boldsymbol{z}_i$ とする．

3.4 Step 3: 埋め込み表現予測

Step 3 では，埋め込み表現予測を行う．

$k, k+1$ 番目の埋め込み表現 $(\boldsymbol{z}_k, \boldsymbol{z}_{k+1})$ を予測器に入力し，続く $k+2$ 番目の埋め込み表現 $(\boldsymbol{z}_{k+2})$ を予測する．k を 1 から $\lfloor \frac{L}{W} \rfloor - 2$ まで 1 ずつ増やすことで，次の予測した埋め込み表現列 $\boldsymbol{Z}_{\text{pred}}$ を得る．

$$\boldsymbol{Z}_{\text{pred}} = \{\hat{\boldsymbol{z}}_3, \hat{\boldsymbol{z}}_4, ..., \hat{\boldsymbol{z}}_{\lfloor \frac{L}{W} \rfloor - 1}, \hat{\boldsymbol{z}}_{\lfloor \frac{L}{W} \rfloor}\} \tag{8}$$

$$\hat{\boldsymbol{z}}_{k+2} = Predictor(\boldsymbol{z}_k, \boldsymbol{z}_{k+1}) \tag{9}$$

ここで，予測した i 番目の埋め込み表現を $\hat{\boldsymbol{z}}_i$ とする．

3.5 Step 4: 異常動作検知

Step 4 では，Step 2 で得た埋め込み表現列 $\boldsymbol{Z}$ と Step 3 で得た予測した埋め込み表現列 $\boldsymbol{Z}_{\text{pred}}$ を比較することで異常動作を検知する．

まず，埋め込み表現 $\boldsymbol{z}_i, \hat{\boldsymbol{z}}_i$ を大きさ 1 のベクトルとしてそれぞれ正規化を施し，$\tilde{\boldsymbol{z}}_i, \tilde{\hat{\boldsymbol{z}}}_i$ とする．そして，2 つの埋め込み表現間の距離を式 (10) のように定義し，2 つの埋め込み表現間の距離を算出する．

$$Distance(\tilde{\boldsymbol{z}}_i, \tilde{\hat{\boldsymbol{z}}}_i) = \frac{1}{m} \sum_{j=1}^{m} \left(\tilde{\boldsymbol{z}}_i[j] - \tilde{\hat{\boldsymbol{z}}}_i[j] \right)^2 \tag{10}$$

ここで，$\tilde{\boldsymbol{z}}_i[j]$ は $\tilde{\boldsymbol{z}}_i$ の j 番目の成分，$\tilde{\hat{\boldsymbol{z}}}_i[j]$ は $\tilde{\hat{\boldsymbol{z}}}_i$ の j 番目の成分とする．

i を 3 から $\lfloor \frac{L}{W} \rfloor$ まで 1 ずつ増やすことで，埋め込み表現の予測誤差を算出する．埋め込み表現間の距離は定常状態電力の影響を受けず，アプリケーション動作の差異のみを反映するため，より正確な異常動作検知が可能となる．この予測誤差があらかじめ設定された閾値を上回る場合に該当部分を異常動作として検出する．

4. 埋め込み表現を用いた異常動作検知手法の改良

本章では，3 章で説明した従来手法を改良した手法を提案する．提案手法の構成を図 2 に示す．

4.1 エンコーダに対する CNN の導入

従来手法では，エンコーダを 2 層の全結合層のみで構成していた．全結合層のみだと，部分波形を 1 つのベクトルでしか見ることができず，消費電力波形の特徴を捉えきれない可能性がある．

提案手法のエンコーダは表 1 に示すように 2 層の CNN 層，2 層の Maxpool 層，2 層の全結合層から構成される．エンコーダに CNN を用いることで，複数のフィルタが局

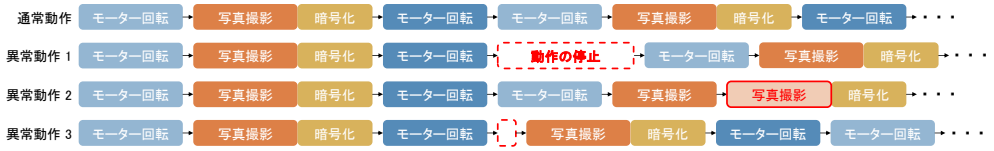


図 3: アプリケーションの動作の遷移.

表 1: CNN ベースエンコーダの入出力形状.

	レイヤー名	入力形状	出力形状
Encoder	Conv1D_1	$W \times 1$	$W \times 12$
	Maxpool_1	$W \times 12$	$\frac{W}{2} \times 12$
	Conv1D_2	$\frac{W}{2} \times 12$	$\frac{W}{2} \times 12$
	Maxpool_2	$\frac{W}{2} \times 12$	$\frac{W}{4} \times 12$
	Dense_1	$\frac{W}{4} \times 12$	30
	Dense_2	30	5
Decoder	Dense_3	5	30
	Dense_4	30	$\frac{W}{4} \times 12$
	Upsample_1	$\frac{W}{4} \times 12$	$\frac{W}{2} \times 12$
	Conv1DTranspose_1	$\frac{W}{2} \times 12$	$\frac{W}{2} \times 12$
	Upsample_2	$\frac{W}{2} \times 12$	$W \times 12$
	Conv1DTranspose_2	$W \times 12$	$W \times 1$

表 2: Transformer ベース予測器の入出力形状.

レイヤー名	入力形状	出力形状
Positional-Encoding	3×5	3×5
Multi-Head-Attention	3×5	3×5
Dropout_1	3×5	3×5
Add_1	3×5	3×5
Layer-Normalization_1	3×5	3×5
Dense_1	3×5	3×32
Dense_2	3×32	3×5
Dropout_2	3×5	3×5
Add_2	3×5	3×5
Layer-Normalization_2	3×5	3×5
Global-Average-Pooling1D	3×5	5
Dense_3	5	32
Dense_4	32	1×5

所的な部分波形の特徴を捉えられ，抽出される表現は，より意味のある表現になると考えられる．

また，予備実験の結果，埋め込み次元数は 5 が最適と分かったため，埋め込み次元数は 5 としている．

4.2 予測器に対する Transformer の応用

従来手法では，予測器を 1 層の LSTM (Long-Short-Term Memory) 層と 1 層の全結合層のみで構成していた．しかし，この単純な構造では埋め込み表現間の複雑な関係性や時系列の文脈を明示的にモデル化する機構が不足しており，予測精度に限界がある．

提案手法の予測器は表 2 に示すように，Transformer アーキテクチャをベースに構成される．Transformer は注意機構ベースのモデルであり，近年では時系列解析においてもその有効性が実証されている．埋め込み表現の系列に対して Transformer の位置エンコーディングと自己注意機構を適用することで，自然言語の文脈を処理するように，それぞれの埋め込み表現間の関係性や意味的つながりを捉えて予測することが可能となる．

表 3: 測定デバイス.

デバイス名	型番	使用用途
オシロスコープ	Tektronix MSO64B	電流，電圧測定
電流プローブ	Tektronix TCP0030A	電流測定
電源装置	KEITHLEY 2280S-32-6	IoT デバイスへの給電

5. 評価実験

本章では提案手法を実際の IoT デバイスに適用した結果を示す．

5.1 対象デバイス

本稿では，異常動作検知対象の IoT デバイスとして，Raspberry Pi4 上に，4 種類の異なる動作を実行するアプリケーションを実装した．図 3 にアプリケーションの動作の遷移を示し，それぞれの動作を次に示す:

通常動作: まず初めにモータを回す．その後写真を撮影し，その画像を暗号化する．そして，再びモータを回す．

異常動作 1: 一定時間 IoT デバイスのアプリケーションを停止させる．

異常動作 2: 通常動作の一連の動作の中で 2 度目の写真撮影が行われる (通常動作では写真撮影は 1 度しか行われない)．

異常動作 3: まず初めに，写真を撮影し，その画像を暗号化する．そして，モータを回す (通常動作における最初のモータの回転をスキップする)．

Raspberry Pi4 上で動作する一連のアプリケーションプログラムは，通常動作を繰り返し実行しており，異常動作は低確率で通常動作の代わりに実行される．

5.2 実験環境

提案手法を Intel Xeon Gold 6238 CPU と 755GB のメモリを搭載したコンピュータに Python3.6.9 で実装し，実験を行った．表 3 に使用した測定デバイスを示す．

評価実験のために，5.1 節で説明した IoT デバイスを動作させ，89 個の総消費電力を測定した．1 つの電力波形はサンプリングレート 125kHz で 10 秒間測定したものである．測定データを 1250 おきに抽出し，全ての電力波形は

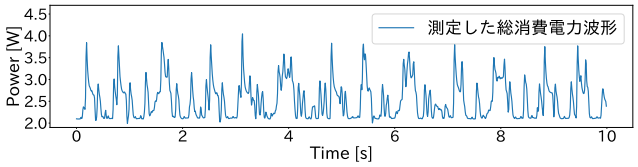


図 4: 測定した総消費電力波形の例.

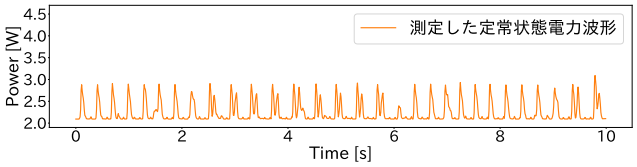
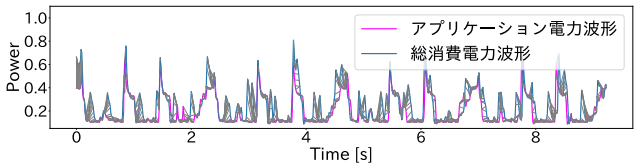
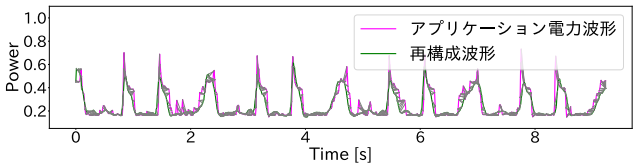


図 5: 測定した定常状態電力波形の例.



(a) 総消費電力波形とアプリケーション電力波形 ($DTW_{\mathrm{total-app}}$).



(b) 再構成波形とアプリケーション電力波形 ($DTW_{\mathrm{recon-app}}$).

図 6: DTW 距離の算出の様子.

表 4: DTW 距離の算出結果.

番号	$DTW_{\mathrm{total-app}}$	$DTW_{\mathrm{recon-app}}$	減少率
1	0.0341	0.0144	57.8 %
2	0.0344	0.0150	56.4 %
3	0.0282	0.0145	48.6 %
4	0.0334	0.0142	57.5 %
5	0.0288	0.0145	49.7 %
6	0.0311	0.0147	52.7 %
7	0.0320	0.0149	53.4 %
8	0.0328	0.0145	55.8 %
9	0.0347	0.0142	59.1 %
10	0.0328	0.0145	55.8 %
平均			54.7 %

1000 個の電力データ ($L = 1000$) を持つものとする．測定した総消費電力波形の例を図 4 に示す．また，アプリケーションを動作させずに測定した定常状態電力波形の例を図 5 に示す．図 5 に示すように，複数の定常状態があり，約 4 秒ごとに切り替わっているのが分かる．

89 個のデータのうち，66 個を学習用，23 個を検証用として用いる．学習用 66 個のデータには，異常動作 1，異常動作 2，異常動作 3 がそれぞれ 1 つずつ含まれている．検証用 28 個のデータにも，異常動作 1，異常動作 2，異常動作 3 がそれぞれ 1 つずつ含まれている．この 3 個の異常動作を検知するのが目標となる．

5.3 エンコーダの準備

学習用 66 個のデータを用いてエンコーダの準備を行った．

5.3.1 エンコーダの学習

66 個のデータからウィンドウサイズ $W = 84$ でスライディング形式にて部分波形を抽出し，オートエンコーダの学習を行った．オートエンコーダの機械学習構造を表 1 に示す．オートエンコーダは主に CNN で構成され，長さ 84 の消費電力波形を 5 次元の埋め込み表現に変換する．損失関数としては，オートエンコーダの入力と出力の MSE を利用し，学習回数は 100 とした．学習は 2 分程度で終了した．

5.3.2 エンコーダの検証

3.1.1 項において，エンコーダが抽出する特徴量は比較的消費電力の大きいアプリケーションの動作に起因する電力波形の特徴を捉えていると述べた．本項では，この仮説を実験により検証する．

学習したエンコーダによって抽出される表現が，アプリケーションによる成分の特徴を有することを検証するために，①総消費電力波形，②再構成波形，③アプリケーション電力波形の 3 つの波形を用意し，次の 2 つの条件下で DTW (Dynamic Time Warping) [22] 距離を算出した．

(1) ①総消費電力波形と③アプリケーション電力波形の DTW 距離 ($DTW_{\mathrm{total-app}}$)

(2) ②再構成波形と③アプリケーション電力波形の DTW 距離 ($DTW_{\mathrm{reocn-app}}$)

DTW 距離とは時系列データ同士の距離を表し，時系列同士の長さや周期が違っても適切に距離を算出できる特徴がある．

定常状態電力は常に発生するため，通常はアプリケーション電力のみは測定できないが，今回は定常動作を実行するハードウェアの制御インターフェースを切断することで定常動作を停止させ，③アプリケーション電力波形を測定する．②再構成波形は，測定した①総消費電力波形をエンコーダを用いて一度 5 次元の埋め込み表現に変換し，デコーダによって再び波形に戻したものである．

DTW 距離は，対応する点同士の MAE (Mean Absolute Error) で表現され，2 つの波形の位相を相互相関によってできるだけ揃えてから算出する．$DTW_{\mathrm{total-app}} > DTW_{\mathrm{recon-app}}$ となれば，再構成波形の方がアプリケーション電力波形との類似度が高いことになり，エンコーダによって抽出された表現がアプリケーションによる成分の特徴を持つと言える．

DTW 距離の算出結果を表 4 に示す．DTW 距離は検証用データに含まれる 10 個の波形を用いて算出した．また，対応する点の例を，図 6a と図 6b に示す．図 6a，図 6b において，灰色の線で繋がれたデータポイント同士の MAE

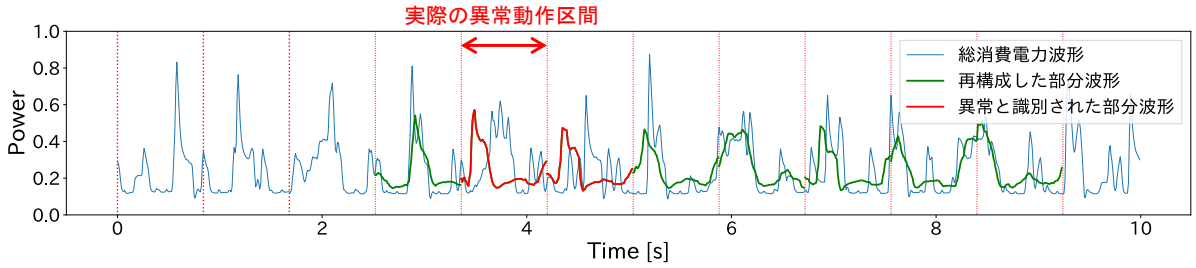


図 7: 提案手法による埋め込み表現予測の結果と異常動作検知結果 (異常動作 3 を含む場合).

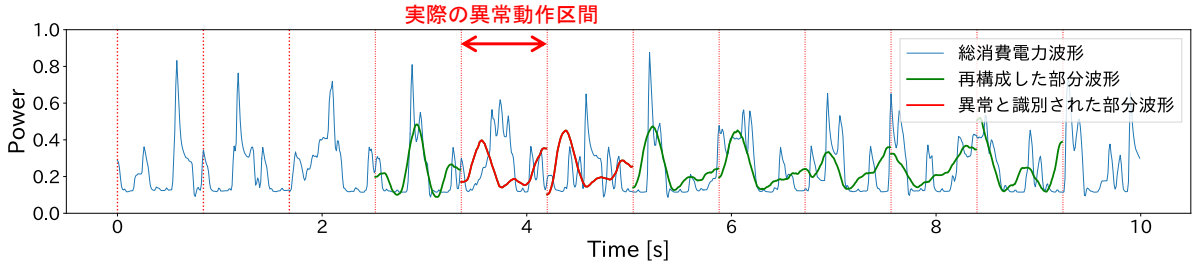


図 8: 従来手法 [16] による埋め込み表現予測の結果と異常動作検知結果 (異常動作 3 を含む場合).

が DTW 距離となる.

表 4 に示すように，検証用データの 10 個の波形において，平均 54%以上 DTW 距離が減少したことから，再構成波形は総消費電力波形に比べてアプリケーション電力波形に近いと分かる．再構成波形がアプリケーション電力波形に近いということは，エンコーダにより抽出される 5 次元の特徴量は，定常状態電力波形の影響が除かれた，アプリケーション電力波形の特徴を持つと言える.

以上より，3.1.1 項で述べたことを実験的に検証できた.

5.4 予測器の準備

続いて，予測器の準備を行なった．66 個のデータから長さ $W = 84$ ごとに部分波形を抽出し，エンコーダを用いて埋め込み表現へ変換した．1 個の波形あたり，$\lfloor 1000/84 \rfloor = 11$ 個の埋め込み表現が得られる．予測器の機械学習構造を表 2 に示す．予測器は Transformer をベースに構成され，3 個の埋め込み表現 $(\boldsymbol{z}_k, \boldsymbol{z}_{k+1}, \boldsymbol{z}_{k+2})$ を入力として受け取り，続く 1 個の埋め込み表現 $(\boldsymbol{z}_{k+3})$ を予測する．損失関数としては，予測器による出力 $\hat{\boldsymbol{z}}_{k+3}$ と $\boldsymbol{z}_{k+3}$ の MSE を利用し，学習回数は 200 とした．学習は 2 分程度で終了した.

5.5 異常動作検知結果

本実験では，Step 0 で使用した検証用データ 23 個を異常動作検知対象の消費電力データとする.

提案手法における，埋め込み表現の予測結果と異常動作検知結果を図 7 に示す．図 7 において，緑色と赤色の部分波形は，予測器によって出力された埋め込み表現を Step 0 で得たオートエンコーダのデコーダ部分に入力して再構成したものである．提案手法は総消費電力波形からアプリケーションの動作に起因する電力波形の特徴を抽出し，正しく異常動作を検知できているのが分かる.

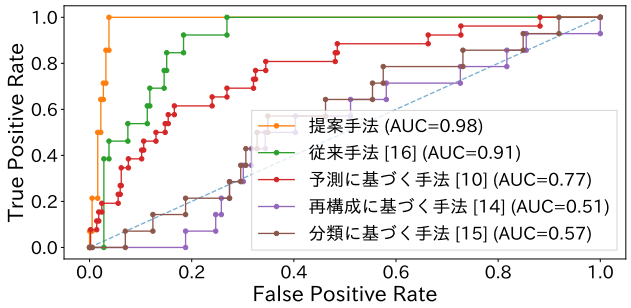


図 9: 提案手法と比較手法の ROC 曲線.

また，図 8 に従来手法による埋め込み表現予測の結果と異常動作検知結果を示す．視覚的な比較ではあるが，提案手法の方がアプリケーション電力波形の特徴をより効果的に抽出，予測できているのが分かる．さらに，5.3.2 項と同様の手法で DTW 距離を算出した結果，検証用データの 10 個の波形において平均 8%程度の距離減少にとどまった．この結果は，提案手法が異常動作の影響を受けるアプリケーション電力の特徴をより捉えられていることを示している.

5.6 定量的評価

提案手法の有効性を確認するために，提案手法の他に，従来手法 [16]，予測に基づく手法 [10]，再構成に基づく手法 [14]，分類に基づく手法 [15] をそれぞれ新たに用意したデータセットに適用し，ROC 曲線を作成し，AUC を算出した.

手法の有効性をより適切に評価するため，新たに用意したデータセットは 25 個のデータで構成され，異常動作 1 が 4 個，異常動作 2 と異常動作 3 がそれぞれ 5 個ずつ含まれている.

得られた ROC 曲線と AUC を図 9 に示す．図 9 より，

提案手法が最も高い AUC を示し，従来手法と比較して検知精度が 0.07 向上していることが確認できる．従来手法は単純な機械学習構造にも関わらず高い検知精度を有しており，これは埋め込み表現に基づく異常動作検知フレームワークの有効性を裏付けていると考えられる．提案手法における更なる精度向上は，エンコーダと予測器の機械学習構造の改良に起因すると考えられる．

6. おわりに

本稿では，埋め込み表現を用いた異常動作検知手法 [16] を改良し，消費電力波形を埋め込み表現に変換するエンコーダと埋め込み表現予測を行う予測器の機械学習構造を改良することで，表現力の高い埋め込みと，異常動作検知精度の向上を実現した手法を提案した．実験の結果，従来手法よりも表現力の高い埋め込みを実現し，異常動作検知精度の向上も達成した．

今後の課題としては，提案手法を多様なデバイスに適用することで，その有効性を示すことが挙げられる．

謝辞

本研究成果は，一部，国立研究開発法人情報通信研究機構 (NICT) の委託研究 (JPJ012368C08101) により得た．

参考文献

[1] I. Ahmad, M. S. Niazy, R. A. Ziar, and S. Khan, “Survey on iot: security threats and applications,” *Journal of Robotics and Control (JRC)*, vol. 2, no. 1, pp. 42–46, 2021.

[2] P. K. Sadhu, V. P. Yanambaka, and A. Abdelgawad, “Internet of things: Security and solutions survey,” *Sensors*, vol. 22, no. 19, p. 7433, 2022.

[3] A. Qureshi, M. Marvi, J. A. Shamsi, and A. Aijaz, “euf: A framework for detecting over-the-air malicious updates in autonomous vehicles,” *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 8, pp. 5456–5467, 2022.

[4] A. P. Sayakkara and N.-A. Le-Khac, “Electromagnetic side-channel analysis for iot forensics: Challenges, framework, and datasets,” *Ieee Access*, vol. 9, pp. 113585–113598, 2021.

[5] J. Wen, C. Yan, X. Ji, and W. Xu, “Power iot system security monitoring based on power consumption side channel information,” in *2022 5th International Conference on Electronics and Electrical Engineering Technology (EEET)*, pp. 59–65, IEEE, 2022.

[6] C. He, D. Lei, H. Wu, L. Cheng, G. Yan, and Q. Huang, “A side-channel hardware trojan detection method based on fuzzy c-means clustering and fusion distance algorithms,” *IEEE Internet of Things Journal*, 2023.

[7] K. I. Gubbi, B. Saber Latibari, A. Srikanth, T. Sheaves, S. A. Beheshti-Shirazi, S. M. PD, S. Rafatirad, A. Sasan, H. Homayoun, and S. Salehi, “Hardware trojan detection using machine learning: A tutorial,” *ACM Transactions on Embedded Computing Systems*, vol. 22, no. 3, pp. 1–26, 2023.

[8] S. Alnegheimish, L. Nguyen, L. Berti-Equille, and K. Veeramachaneni, “Can large language models be anomaly detectors for time series?,” in *2024 IEEE 11th International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 1–10, 2024.

[9] M. Zhao, H. Peng, L. Li, and Y. Ren, “Graph attention network and informer for multivariate time series anomaly detection,” *Sensors*, vol. 24, no. 5, p. 1522, 2024.

[10] R. Eda and N. Togawa, “Anomalous iot behavior detection by lstm-based power waveform prediction,” in *Proceedings of the 10th International Conference on Internet of Things, Big Data and Security - IoTBDS*, pp. 345–352, INSTICC, SciTePress, 2025.

[11] S. Maitra, S. Kundu, and A. Shankar, “A real-time anomaly detection using convolutional autoencoder with dynamic threshold,” *arXiv preprint arXiv:2404.04311*, 2024.

[12] S. Yi, S. Zheng, S. Yang, G. Zhou, and J. He, “A model fine-tuning approach for robust anomaly detection and isolation in multi-sensor system of nuclear power plants,” *Annals of Nuclear Energy*, vol. 205, p. 110557, 2024.

[13] X. Zhang, S. Xu, H. Chen, Z. Chen, F. Zhuang, H. Xiong, and D. Yu, “Rethinking robust multivariate time series anomaly detection: A hierarchical spatio-temporal variational perspective,” *IEEE Transactions on Knowledge and Data Engineering*, 2024.

[14] A.-T. W. K. Fahmi, K. R. Kashyzadeh, and S. Ghorbani, “Fault detection in the gas turbine of the kirkuk power plant: An anomaly detection approach using dlstm-autoencoder,” *Engineering Failure Analysis*, vol. 160, p. 108213, 2024.

[15] R. EDA, K. HISAFURU, and N. TOGAWA, “Genpower2: Improved anomaly detection in iot devices utilizing generated power waveforms,” *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, p. 2024EAP1145, 2025.

[16] 江田琉聖 and 戸川望, “時系列埋め込み表現を用いた iot デバイス異常動作検知手法,” in **信学技報**, vol. 125 of *CAS2025-14, VLD2025-14, MSS2025-14*, (青森), pp. 69–74, 6 月 2025. 2025 年 6 月 19 日 (木)-6 月 20 日 (金) 八戸工大サテライトキャンパス (CAS, VLD, MSS).

[17] K. Nimmy, M. Dilraj, S. Sankaran, and K. Achuthan, “Leveraging power consumption for anomaly detection on iot devices in smart homes,” *Journal of Ambient Intelligence and Humanized Computing*, vol. 14, no. 10, pp. 14045–14056, 2023.

[18] J. Ge, J. Rui, H. Ma, B. Li, and Y. He, “Contextual insight: Detecting abnormal device behaviors in iot systems,” in *2024 27th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pp. 1268–1273, IEEE, 2024.

[19] L. H. Koopmans, *The spectral analysis of time series*. Elsevier, 1995.

[20] A. Ermshaus, P. Schäfer, and U. Leser, “Window size selection in unsupervised time series analytics: A review and benchmark,” in *Proceedings of International Workshop on Advanced Analytics and Learning on Temporal Data*, pp. 83–101, Springer, 2023.

[21] S. Voloshynovskiy, M. Kondah, S. Rezaeifar, O. Taran, T. Holotyak, and D. J. Rezende, “Information bottleneck through variational glasses,” *arXiv preprint arXiv:1912.00830*, 2019.

[22] H. Sakoe and S. Chiba, “Dynamic programming algorithm optimization for spoken word recognition,” *IEEE transactions on acoustics, speech, and signal processing*, vol. 26, no. 1, pp. 43–49, 2003.