キーストロークデータを用いたプログラミング経験判定 AI モデルの構築

滝澤菜々子† 佐藤 美唯† 高野 志歩†† 小原百々雅†† 田村 みゆ††
倉光 君郎†

† 日本女子大学理学部数物情報科学科 〒 112–8681 東京都文京区目白台 2-8-1
†† 日本女子大学大学院理学研究科数理・物性構造科学専攻 〒 112–8681 東京都文京区目白台 2-8-1
E-mail: †{m1916047tn}@ug.jwu.ac.jp, ††kuramitsuk@fc.jwu.ac.jp

あらまし キーストロークデータとは，パソコンキーボードの打鍵速度や打ち方，リズムを表す情報のことである．キーボードの打ち方の癖は人によって全く異なり，近年，機械学習の発展により，セキュリティや生体認証の分野でキーストローク解析が利用されている．本研究では，プログラミング経験が豊富な人はプログラミング言語特有の関数名や特殊文字を打ち慣れているという仮定に基づき，キーストロークデータの解析をすることでその人がどの程度のプログラミング経験があるかを判定する機械学習モデルを実現する．プログラミング初学者からエンジニアまでのキーストロークデータを収集し，Random Forest と Transformer を用いてそれぞれ分類モデルを構築した．本発表では，構築した判定 AI モデルの精度を報告し，判定システムの試作を紹介する．

キーワード キーストローク解析，機械学習，プログラミング経験

1 はじめに

キーストロークデータとは，パソコンのキーボードで文字を入力するときの打鍵速度，打ったキーの種類，リズムの情報のことである．パソコンキーボードの打ち方は人によって全く異なり [1]，キーストロークデータには個人の特徴がよく表れる．この性質は生体認証や感情分析，教育分野などのさまざまな分野で活用されている．また，キーストロークデータの収集には特別なハードウェアを用意せずに行えることから，キーストロークの個人特性を利用することは現代の情報社会において大きな役割を担っていくと考えられる．

本研究では，プログラミングコードをタイピングした時のキーストロークデータからその人がどの程度のプログラミング経験のユーザであるかを判定することを目的とする．プログラミング言語は，普段タイピングする自然言語には含まれないような関数名や特殊文字を多く含む．そのため，プログラミング経験が豊富な人はプログラミング言語特有の文字を打ち慣れているなどの特徴がキーストローク特徴として表れるのではないかと考えた．キーストロークデータにユーザのプログラミング経験が影響することが確認できれば，コーディングテストなどを実施することなくタイピングをするだけでユーザのプログラミング経験や技量を図ることができる．キーストロークデータからプログラミング経験の判定を実現するために，我々は，様々なプログラミング経験の人から実際にキーストロークデータを収集し，そのデータを用いてプログラミング経験判定 AI モデルを構築した．本論文では，実際に構築した判定 AI モデルの精度を報告し，試作した判定システムを紹介する．

本論文の構成は以下の通りである．2 節では，提案手法について述べる．3 節では，本研究において使用した 2 種類の機械学習モデルついて述べる．4 節では，実験方法と評価結果，および判定システムについて紹介する．5 節では，関連研究を述べる．6 節で本論文をまとめる．

図 1 提案手法

2 提案手法

本節では，キーストロークからその人のプログラミング経験を判定するための手法として，機械学習モデルを用いることを提案する．提案手法を図 1 に示す．プログラミング初心者からエンジニアまでの様々なプログラミング経験の人から収集したキーストロークデータを機械学習モデルに学習させることで，プログラミング経験判定 AI モデルを構築する．

2.1 キーストロークデータ

本研究で利用するキーストロークデータについて詳細を説明する．本研究では，キーストロークの癖を表すデータとして，打ったキーの種類と p-p 時間（あるキーを押してから次のキーを押すまでの時間）の組の並びをキーストロークデータとする．具体的には，図 1 に示すような，数字と文字の並びからなるデータである．時間の単位は，ms（ミリ秒）である．

3　機械学習モデルによる分類予測

機械学習とは，大量の学習データからパターンやルールをコンピュータに発見させ，ある課題において分類や予測を行う技術のことである．本研究では，学習データに正解を与えた状態で学習させることで，未知のデータを入力しても分類の推論ができる「教師あり学習」を行う．我々は，プログラミング経験判定 AI モデルの構築のために，機械学習モデルのひとつである RandomForest [2] と Transformer [3] を用いた．

3.1　RandomForest

RandomForest は，2001 年に Breiman によって提案された機械学習アルゴリズムである．学習データをランダムにサンプリングして複数の決定木を作成し，それぞれの出力結果の多数決により最終的な出力結果が得られる．さまざまなタスクにおいて汎用性が高く，精度が高いことが特徴である．

本研究では，連続する 2 つのキーにおける p-p 時間の平均を特徴量として RandomForest に学習させる．すなわち，打ったキー 2 個分のみの依存関係に着目している．

3.2　Transformer

Transformer は 2017 年に Vaswani によって発表された，自然言語処理向けの深層学習モデルである．それまで自然言語処理でよく用いられていた再帰や畳み込みを一切使わず，単語間のより広範囲な依存関係を捉えられる Attention 機構のみで構成されていることが特徴である．

Transformer は，自然言語処理のさまざまなタスクにおいてこれまで多くの SoTA を達成してきた．近年，自然言語処理以外の分野（コンピュータビジョン，音声認識など）のタスクにも Transformer が適用されている．例えば，画像分類タスクにおいては，画像を N 個のパッチに分割し，各パッチをトークンと呼ばれる単語のようなものとして捉えることで Transformer ベースモデルへ入力し，分類を行っている．本研究では，キーストロークデータの打ったキーと p-p 時間をそれぞれトークンとして扱い Transformer ベースモデルに入力することで，プログラミング経験の判定を行う．

RandomForest は，打ったキーの順番などは考慮せず，2 つ分のキーの依存関係のみを捉えることに対して，Transformer は打ったキーの順番なども考慮し，キーストロークデータ中のより広範囲な依存関係を捉えられるのではないかと考えている．我々が構築した，Transformer ベースのモデルアーキテクチャを図 2 に示す．

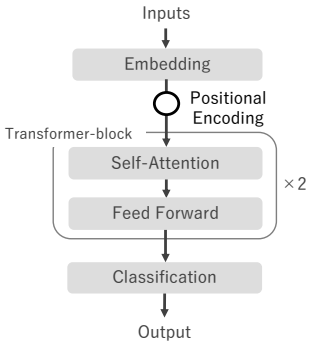

図 2　Transformer ベースモデルのアーキテクチャ

4　実 験 報 告

本節では，3 節で紹介した RandomForest と Transformer に実際に収集したキーストロークデータを学習させ，プログラミング経験判定 AI モデルを構築し，それぞれどの程度正しく判定できたか比較した結果を報告する．

4.1　キーストロークデータの収集

我々は，被験者に指定した python コードを実際にタイピングしてもらい，その際のキーストロークデータをログとして記録した．我々が指定した python コードを図 3 に示す．プログラミング経験による打ち慣れおよび不慣れがより表れるように，「print(“ ”)」や「*」などのプログラミング言語特有の関数名や特殊文字を含んだ文を指定した．

```
print(math.sin(math.pi/2))
print(["oranges","tables"])
print(weight / (height * height))
print(x if x >= y else y)
print(s[0].upper() for s in "abcdefg")
```

図 3　指定した python コード

また，キーストロークデータにはその人のプログラミング経験もセットにして記録した．本実験において，我々は被験者のプログラミング経験を 3 グループに分けている．そのグループの名称と，実際の経験および所属についての説明を表 1 に示す．また，収集できたデータ件数も表 1 に加えて示す．

表 1　プログラミング経験とその説明

名称	説明	件数
未経験	プログラミング経験のない中高生	28 件
初学者	日本女子大学 2 年生（履修授業にて python プログラミングを半年ほど扱っている）	59 件
経験者	プログラミングを 3 年以上または業務で扱っている社会人	45 件

以上，合計 132 件のキーストロークデータを収集することができた．

4.2　学習用データの前処理

本節では，収集したキーストロークデータを機械学習モデルに学習させるために行った前処理について説明する．まず，全てのキーストロークデータにおいて，先頭の「p-p 時間+打っ

たキー」のペアを削除した．これは，データ収集ページを起動してから 1 文字目を打つまでの時間はかなりばらつきがあり，キーストロークの癖には関係のない情報であると考えられるためである．

また，本研究では Transformer に学習させるキーストロークデータに対して 2 つのデータ前処理を行うことでデータの拡張を行なった．深層学習モデルの訓練に必要なデータの量は，機械学習モデルに比べて大量の学習データを必要とする．本実験で収集できたデータの量は小規模であり，モデル本来の性能を発揮できないため，データの拡張を行なった．

1 つ目のデータ前処理方法はスライディングウィンドウ処理である．スライディングウィンドウ処理とは，主に時系列データに用いられる前処理のことであり，1 本の時系列から部分時系列を多数生成し，機械学習により特徴を抽出する方法である [4]．部分時系列の生成は，1 本のデータに対してウィンドウサイズと呼ばれる固定幅で区切り，区切る位置を少しずつスライドさせることで多数生成する．本研究では，ウィンドウサイズ n とした時，n 文字打った分のキーストロークデータが多数生成される．スライディングウィンドウ処理のイメージを図 4 に示す．

本研究では，Transformer に学習させるキーストロークデータを一本の時系列データとみなし，スライディングウィンドウ処理を適用した．

図 4　スライディングウィンドウ処理

2 つ目のデータ前処理方法は，Data Augmentation である．Data Augmentation とは，既存のデータに対して何らかの変更を加えることにより，訓練データの量を水増しする手法である．本研究では，既存のキーストロークデータの p-p 時間に対して，標準偏差と平均値を用いて 1 箇所ずつ変更を加えることでデータを拡張した．4.3 節にて 2 つのデータ拡張方法によるモデルの精度の比較結果を述べる．

4.3 評価結果

本節では，Random Forest，Transformer（スライディングウィンドウ処理），Transformer（Data Augmentation）の 3 つのモデルの予測精度を比較する．評価に用いた評価尺度は以下に示す 4 つである．

- 正解率：予測結果と真の値の一致率.
- 適合率：真と予測したもののうち、実際に真である割合.
- 再現率：実際に真であるもののうち，正しく真と予測できた割合.
- F 値：再現率と適合率の調和平均.

再現率は，実際に真であるが偽と予測したというケースをエラーとして重視したい場合に向いている．例えば，実際は初学者であるのに，上級者と判定されてしまった場合などが多い時，再現率は低くなる．再現率は適合率とトレードオフの関係である．F 値は，例えば再現率と適合率のどちらかがかなり低い時に，値が低くなる．

構築した 3 つのモデルによる判定 AI モデルの評価尺度をまとめた表を，表 2，表 3，表 4 にまとめる．また，正解率は，RandomForest は 0.80，Data Augmentation による Transformer は 0.70，スライディングウィンドウ処理による Transformer は 0.92 であった．

.

表 2　RandomForest モデルの評価結果

	未経験	初学者	経験者
適合率	1.00	0.76	0.81
再現率	0.38	0.89	0.93
F 値	0.55	0.93	0.87

.

表 3　Transformer モデルの評価結果（Data Augmentation）

	未経験	初学者	経験者
適合率	0.00	0.64	0.88
再現率	0.00	0.78	1.00
F 値	0.00	0.70	0.93

.

表 4　Transformer モデルの評価結果（スライディングウィンドウ処理）

	未経験	初学者	経験者
適合率	1.00	0.88	0.91
再現率	0.71	1.00	1.00
F 値	0.83	0.93	0.95

Random Forest の評価尺度は全体的に高い結果だったが，未経験者の再現率が 0.38 であるように，未経験者のキーストローク特徴をうまく捉えられていなかった．Data Augmentation による Transformer は，未経験者のキーストローク特徴を全く捉えられていない結果となった．3 つのモデルを比較すると，スライディングウィンドウ処理をした Transformer による判定 AI モデルが，どのラベルにおいても最も予測精度が高かった．

4.4 判定システム

本節では，4.3 節にて構築した判定 AI モデルを用いて試作した，プログラミング経験をリアルタイムに判定できる判定システムを紹介する．判定 AI モデルは，Random Forest によるものを用いて試作した．システムの作成には，Streamlit という Python 言語によって簡易的に Web アプリケーションが作成できる Python フレームワークを用いた．判定システムの UI を図 5 に示す．

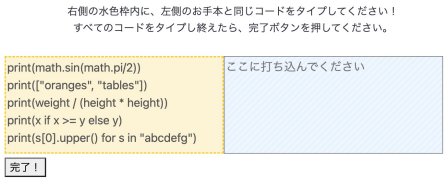

図 5　試作した判定システムの UI

左枠内に示されている Python コードをお手本に右枠内にタイピングすると，システム内部で JavaScript のキーイベントが動作し打ったキーの種類と p-p 時間が記録される．タイピングし終え完了ボタンを押すと，キーストロークデータが判定 AI モデルに入力され，プログラミング経験の判定結果が画面に出力される．

5 関 連 研 究

キーストロークは，ユーザがそれぞれ異なる挙動を示す点が認証技術の特性として望ましく，行動生体認証 [5], [6], [7] などのセキュリティ分野で幅広く研究されてきた．近年は，機械学習技術 [8], [9] の目覚ましい発展により，キーストロークを用いた行動生動認証の実用度も大幅に高まっている．

キーストロークの活用は，セキュリティ分野だけでなく，感情分析など応用も広がっている．教育分野 [10], [11] では，学生の学習成果などを把握するために応用される．

6 む す び に

本論文では，キーストロークデータからユーザのプログラミング経験を判定するために，機械学習モデルを用いてプログラミング経験判定 AI モデルを構築することを提案した．実際には，RandomForest と Transformer を用いて判定 AI モデルを構築し，それぞれの精度の比較結果を報告した．スライディングウィンドウ処理をしたデータによる Transformer が一番精度が高く，正解率は 0.92 であった．また，構築した判定 AI モデルを用いて試作した判定システムの紹介をした．判定システムは，指定された Python コードをタイピングしたのち判定結果が出力される機能のみの試作段階であるが，別のシステムと統合することで有用性が増すと考えている．例えば，我々の研究室で開発しているプログラミング学習支援 AI「KOGI」が挙げられる．KOGI とは，コーディング中にエラーが発生するとエラーの原因や解決策を提示するチャットボット形式の AI であるが，本研究で作成した判定システムと統合することで，コーディングテストなどを用いることなくユーザのプログラミングの技量を推定することができ，それに応じた学習支援内容を提供するなどの応用が期待できる．

今後は，試作した判定システムを実際に使うことで挙動を確認し，さらなるデータの収集およびデータのラベル付けの再検討などをすることで判定システムの有用性を高めていきたい．

文　　献

[1] Anil K Jain, Arun Ross, and Salil Prabhakar. An introduction to biometric recognition. *IEEE Transactions on circuits and systems for video technology*, 14(1):4–20, 2004.

[2] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[4] 井手剛. 部分時系列クラスタリングの理論的基礎. In **人工知能学会全国大会論文集 第** *20* **回** *(2006)*, pages 94–94. 一般社団法人 人工知能学会, 2006.

[5] Francesco Bergadano, Daniele Gunetti, and Claudia Picardi. User authentication through keystroke dynamics. *ACM Trans. Inf. Syst. Secur.*, 5(4):367–397, nov 2002.

[6] Alaa Darabseh and Akbar Siami Namin. Keystroke active authentications based on most frequently used words. page 49–54, 2015.

[7] Sohail Habib, Hassan Khan, Andrew Hamilton-Wright, and Urs Hengartner. Revisiting the security of biometric authentication systems against statistical attacks. *ACM Trans. Priv. Secur.*, nov 2022. Just Accepted.

[8] Sowndarya Krishnamoorthy, Luis Rueda, Sherif Saad, and Haytham Elmiligi. Identification of user behavioral biometrics for authentication using keystroke dynamics and machine learning. In *Proceedings of the 2018 2nd International Conference on Biometric Engineering and Applications*, ICBEA '18, page 50–57, New York, NY, USA, 2018. Association for Computing Machinery.

[9] Tanapat Anusas-amornkul. Strengthening password authentication using keystroke dynamics and smartphone sensors. In *Proceedings of the 9th International Conference on Information Communication and Management*, ICICM 2019, page 70–74, New York, NY, USA, 2019. Association for Computing Machinery.

[10] John Edwards, Juho Leinonen, and Arto Hellas. A study of keystroke data in two contexts: Written language and programming language influence predictability of learning outcomes. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, SIGCSE '20, page 413–419, New York, NY, USA, 2020. Association for Computing Machinery.

[11] Aythami Morales and Julian Fierrez. Keystroke biometrics for student authentication: A case study. In *Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education*, ITiCSE '15, page 337, New York, NY, USA, 2015. Association for Computing Machinery.