作業履歴中の主要な作業に着目した作業目的予測

大橋 亮太[1]　上野 秀剛[1]　門田 暁人[2]

概要：本稿はソフトウェア開発者の能力向上・プロセス改善手法である PSP における作業履歴の記録と作業目的に基づいた履歴の分析を支援するために，作業者の作業履歴から作業目的を予測する手法を提案する．提案手法は作業目的が頻繁に変化しないという仮説の元，予測対象となる作業 (Task) の前後に行われた作業の特性から作業目的 (Aim) を推定する．連続する Task のうち，打鍵数やクリック数が多い Task をその Aim における主要 Task とし，特性として使用する．RandomForest による予測モデルを構築し実験した結果，提案手法は時系列情報を用いない場合に比べ，高い精度で作業を予測できた．特に，前後の作業を多く見るほど予測精度の向上が見られた．

1. はじめに

ソフトウェア開発者の能力向上やプロセス改善を目的として設計された Personal Software Process (PSP) が提唱されている [1]．PSP は開発者の作業履歴を記録し，実装やテスト，設計，会議といった個々の開発作業にどれだけの時間を費やしているか分析し, 効率の改善やプロセス改善に役立てる手法である．

これまでに PSP における作業履歴の記録を容易にするための支援システムが複数提案されている[*1] [*2] [*3]．これらのシステムは開発者の作業に費やした時間を自動で計測することができるが, どのような作業を現在行っているかという情報に関しては．手動で入力する必要がある．そのため, データの取り忘れや計測に気を取られ作業に集中できないといった問題がある．

一方で PC 上におけるソフトウェアの使用を計測するシステムとして TaskPit や Manic Time がある [2][*4]．これらのツールでは使用してるソフトウェア名，もしくはそれと関連付けられた作業名と作業時間や打鍵数，クリック数を自動的に記録する．計測者はシステムで記録された作業履歴を設計や実装といった開発作業の種類ごとに集計することで個々の開発作業にどれだけの時間を費やしているか分析できる．

このとき，同じ作業であっても，その目的によって異なる種類に集計する必要がある．例えば設計書に対する作業があった場合，設計書を作成するための作業であれば設計に，実装の過程で設計を確認するための作業であれば実装として分類する．開発プロセスを改善するにあたって，作業の目的に基づいた履歴分析が必要になる．このような作業の目的への分類はシステム側では判断できず，計測者が手動で行う必要があるため，計測者に負担がかかることやデータの漏れが発生する問題点がある．

本稿では TaskPit が計測する作業を Task, 開発者が Task を行っている目的を Aim と定義し，Task の Aim を自動で予測する手法を提案する．Task の Aim を自動で予測する事で，開発者の作業の妨げにならない計測が可能になり，正確な記録からプロセス改善が行えるようになる．提案する手法は，Aim は連続する Task の集合であるとみなし，予測対象 Task と前後に実行される Task の特徴から，機械学習アルゴリズムである Random Forests を用いて Aim を予測する．各 Task の特徴として，Task 名，打鍵数やマウスクリックの数を選択した．また，各 Aim にはそれぞれ打鍵数やクリック数が多い Task（主要な Task）があると考え，予測対象 Task 前後の Task に対して，作業量毎に順位付けを行い，特徴として使用する．

2. 関連研究

2.1 PSP 支援システム

PSP とはソフトウェア開発者が開発作業に費やした時間を計測し，分析する事で開発プロセスを改善する手法である．PSP では各作業に必要な時間を事前に見積り，計測し

[1] 奈良工業高等専門学校
Nara National College of Technology
[2] 奈良先端科学技術大学院大学
Nara Institute of Science and Technology

[*1] Process Dashboard, http://www.processdash.com/
[*2] Task Coach Your friendly task manager, http://members.chello.nl/f.niessink/
[*3] SlimTimer Time Tracking without the Timesheet, http://www.slimtimer.com/
[*4] Manic Time, http://www.manictime.com/

たデータから見積りの差異，及びその原因を分析する事でプロセスの問題点を明らかにする．

開発作業に費やす時間を計測するための支援システムとして Process Dashboard，Task Coach，Slim Timer などがある．これらのシステムは PC 上での作業時間を自動で計測できる一方で，遂行中の作業が切り替わった際に手動で入力をする必要がある．そのため，計測データに漏れが発生し，不正確なデータを記録してしまうといった問題がある．

そこで，作業の計測，判別を自動化したシステムとして TaskPit がある．図 1 に TaskPit のスクリーンショットを示す．TaskPit はアクティブになっているソフトウェアのウィンドウ名と作業 (Task) を関連づけ，作業時間や打鍵数，マウスクリック数を記録する．例えば，“MS Word” で 2 分間作業を行った後，“Open Office Writer” において 3 分間作業を行った場合，“書類閲覧編集” という Task を 5 分間行ったとして記録される．これらの Task の判別は，TaskPit の設定でソフトウェアと Task 名の紐付けをしておくことによって可能となっている．図 2 に TaskPit で計測された作業履歴の例を示す．作業履歴には Task 名，開始時刻，終了時刻，左クリック数，右クリック数，打鍵数が含まれる．TaskPit はアクティブなアプリケーションの変化を検出し，自動的に作業履歴を記録する事ができる．

同様に作業の計測を自動化したシステムとして Manic Time がある．Manic Time では使用しているソフトウェア名やウィンドウタイトルの変化を検出し，それらの名前と費やした時間を自動的に記録することができる．しかし，TaskPit や Manic Time では Task の Aim は自動で判別できないため，作業履歴に Aim が記録されない．そのため，プロセスの改善を行う際に，記録された Task 毎の Aim を調査し，どの Aim に対して作業時間を費やしているかに関して分析する必要があるが，手動で行われるため，計測者にかかる負担や Aim の誤分類によるデータの漏れなどが問題となる．本研究ではプロセス改善の支援を目的として，Task の Aim を自動予測する手法を提案する．

2.2 作業予測と分類

PC 上での作業支援や作業計測を目的とした作業予測と分類に関する研究がされている．Michael らは PC 上のアクティブなウィンドウから機械学習を用いて，PC 上での作業の目的を予測する手法を提案している [3]．この手法はアプリケーション名やウィンドウタイトル，ウィンドウ内のコンテンツを機械学習に使用している．Stumpf らは TaskTracker と TaskPredictor といったシステムを開発している [4]．TaskTracker は PC 上での操作を記録し，TaskPredictor は操作履歴から操作目的を予測する．TaskPredictor は操作の種類や時間，Window ID などを特徴として操作目的を予測している．Oliver らは SWISH と

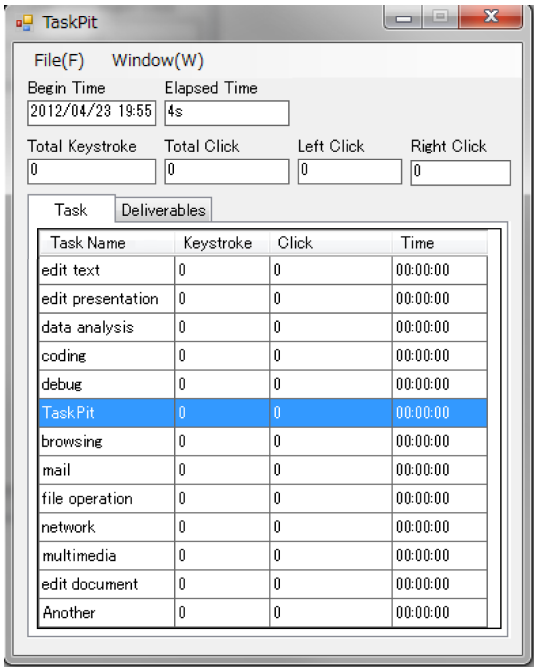

図 1 TaskPit

呼ばれるシステムを提案している [5]．このシステムはウィンドウタイトルやウィンドウの遷移順序を用いて作業目的を予測する．PC 上での作業を円滑にするために開発された TaskPose と呼ばれるシステムがある [6]．このシステムではクラスタリングを用いてウィンドウのグルーピングを行い，関連性のあるウィンドウを互いに近くにレイアウトしたり，重要なウィンドウは大きく表示することで，ウィンドウ間の移動を支援し，作業の円滑化を図っている．

本稿で提案する手法では，TaskPit によって記録された Task の情報から Aim を予測する．また，既存研究では予測対象となる作業の情報を用いて予測を行っているが，本稿では Task の時系列情報を用いて，記録された Task の中から Aim に対する主な Task を判別し，それを特徴とする事で予測精度の向上を目指す．

3. 提案手法

本章では Task と Aim の定義をした後，機械学習アルゴリズムの Random Forests を説明し，時系列情報を用いた提案手法について述べる．

3.1 Task

Task とは 1 つのソフトウェアに対する連続したユーザの操作である．TaskPit はアクティブなアプリケーションに対する操作を監視し，連続した操作を 1 つの Task t として記録する．Task はアクティブ状態になっているウィンドウの起動ファイル名とタイトル名から識別される。同じ Task に対して複数の異なるアプリケーションが利用される場合，それらを区別せず，同じ Task として記録する．また，Web ブラウザのような複数の Task（メールやグルー

Task Name,	Begin Time,	End Time,	Lclick,	Rclick,	Keystroke
design document,	13:25:35,	13:28:48,	11,	2,	98
Req. specification,	13:28:48,	13:29:54,	7,	0,	0
design document,	13:29:54,	13:33:48,	23,	3,	120
desktop,	13:33:48,	13:33:56,	1,	0,	0
design document,	13:33:56,	13:34:48,	4,	0,	75
data analysis,	13:34:48,	13:36:41,	24,	0,	64
desktop,	13:36:41,	13:36:59,	2,	0,	0

図 2 作業履歴例

プウェア）で利用されるアプリであっても，ウィンドウタイトルによって異なる Task として識別する．Task，アプリケーション，ウィンドウタイトルの関係を Extended Backus-Naur Form (EBNF) で示す．

```
<Task> ::= <Application>{|<Application>}
<Application> ::= <exe name> [<window title>]
```

記録される t の特徴を以下に示す．

- Task Name
- Left Clicks
- Right Clicks
- Keystrokes

Task Name は exe のファイル名とタイトルから識別されるタスクの名称を表わす．Left Clicks，Right Clicks，Keystrokes はそれぞれ左，右クリック数，打鍵数を表わす．なお打鍵内容については記録しない．提案手法では，Task 名は実行されていた Task の種類を，左，右クリック数，打鍵数からは Task の作業量を判別するのに用いる．

3.2 Aim

Aim とはユーザが Task を行う目的であり，各 Task t は Aim を達成するために行われる．本研究では，Aim A を連続した t の集合で表す．

$$A = [t_0, t_1, t_2, \cdots, t_n] \tag{1}$$

図 3 に Aim が “設計仕様書作成” である Task 列を示す．図において各四角は 1 つの Task を表し，マウス/キーボードアイコンの数は，ユーザの相対的な作業量を表す．ここで例に示した Aim“設計書作成” は “仕様書閲覧編集” と “設計書閲覧編集” の 3 つの Task から構成されており，“設計書閲覧編集” の作業量は “仕様書閲覧編集” と比べて多いことを示している．

3.3 Random Forests

Random Forests は決定木を弱学習器としたアンサンブル学習を用いる機械学習アルゴリズムである [7]．アンサンブル学習とは，高精度でない分類器を複数組み合わせることで精度を向上させる手法である．Random Forests は以下の手順によって学習を行う．

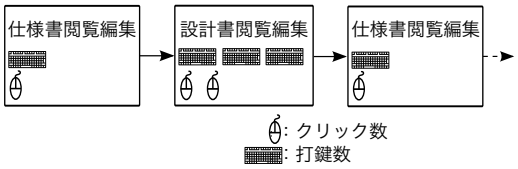

図 3 Aim“設計仕様書作成”

手順 1 与えられたデータセットから $ntree$ 組のブートストラップサンプルを作成する．その際，用いるデータセットの約 3 分の 1 はテスト用データとして取り除かれる．このテスト用データは Out-Of-Bag(OOB) データと呼ばれる．残った 3 分の 2 を学習用データとして用いる．

手順 2 作製したブートストラップ毎に分類木を作成し，木の生成に用いていない OOB データを用いてテストを行う．テスト時の誤り率は OOB 推定値と呼ばれる．分類器の構築を行う際の各分岐ノードは，異なる木を多数生成するため，ランダムに $mtry$ 個の変数をサンプリングし，その中から最も分岐が良い変数を用いる．

手順 3 分類器は，すべてのブートストラップサンプルの OOB 推定値に基づいて多数決を取る．

Random Forests では，組み合わせるアンサンブル学習に用いる分類器の数 $ntree$ と分岐の際に用いる説明変数の数 $mtry$ を調整することができる．本稿では，説明変数の総数を M としたとき，Breiman の推奨する $ntree = 500$，$mtry = \sqrt{M}$ とする [7].

3.4 時系列情報を用いた Aim の予測

ある 1 つの Aim は連続した複数の Task で構成される．Task 列には Aim 毎に，Task の種類や順序，作業量（クリック数や打鍵数）に特徴がある．図 4 に設計，実装，テストを Aim とした Task 列を示す．図は設計と実装では構成する Task の種類が異なる事を示している．また，実装とテストでは構成する Task は同じであるがソースコード閲覧編集 Task で作業量が異なることを示している．Task の種類によっては複数の Aim に分類されうるため，Task

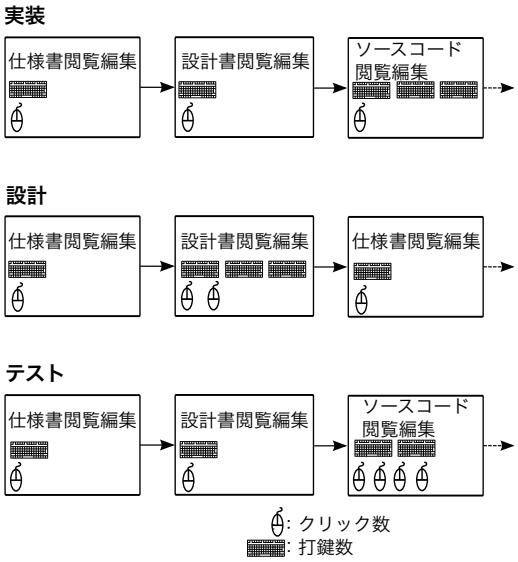

図 4 異なる Aim における Task 列

の名前や作業量などが分かっても，どの Aim に分類されるかまでは分からない．しかし，Task の種類，順序，作業量の特徴を用いれば，前後の Task との関係性から Aim の判別ができると考えられる．提案手法ではこれらの点に着目し，Task の時系列情報から各 Task の Aim を予測する．予測には機械学習アルゴリズムの 1 つである Random Forests を用いる．予測に用いる特徴として Aim の予測対象である Task の情報と，その前後の Task の情報を用いる．予測対象である Task t_p の Aim 予測には Task の集合 $T = [t_{p-x}, t_p, t_{p+x}]$ を用いる．x は考慮する前後の Task の数を示す．予測に用いる特徴として以下の 8 種類の特徴を選択した．

(1) 予測対象 Task 名: $t_p.Name$
(2) 予測対象 Task の左クリック数: $t_p.LClick$
(3) 予測対象 Task の右クリック数: $t_p.RClick$
(4) 予測対象 Task の打鍵数: $t_p.Key$
(5) T の各 Task 名: $t_{p-x}.Name$, ..., $t_p.Name$, ..., $t_{p+x}.Name$
(6) T において左, 右クリック数, 打鍵数それぞれの数値が最も高い Task 名: $T.1stLClick$, $T.1stRClick$, $T.1stkey$
(7) T において左, 右クリック数, 打鍵数それぞれの数値が 2 番目に高い Task 名: $T.2ndLClick$, $T.2ndRClick$, $T.2ndkey$
(8) T において左, 右クリック数, 打鍵数それぞれの数値が 3 番目に高い Task 名: $T.3rdLClick$, $T.3rdRClick$, $T.3rdkey$

提案する手法では Task の Aim が記録された作業履歴に対して，上に挙げた特徴の (5) から (8) を追加し，機械学習のデータセットとして用いる．その後，学習したアルゴリズムを用いて予測対象である作業履歴に記録された Task の Aim を予測する．

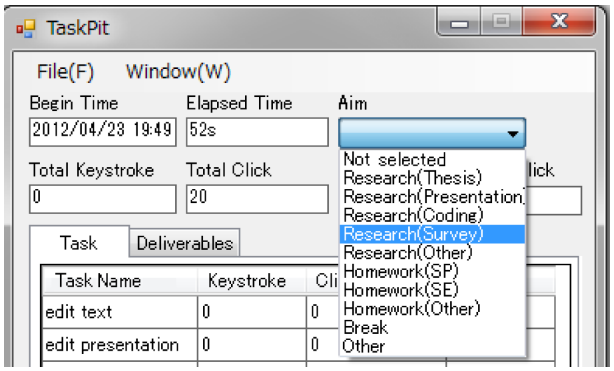

図 5 機能を追加した TaskPit

4. 実験

4.1 実験環境

実験には奈良工業高等専門学校 5 年生の学生 5 人が被験者として参加した．全ての被験者は研究室に配属しており，講義の課題やレポート，研究などを配属先の研究室にある割り当てられた PC で行っている．実験にあたって被験者らが研究室で使用している Windows PC に TaskPit をインストールし，月曜日から金曜日までの五日間の被験者らの Task 及び Aim を計測した．

被験者らの Task の Aim を計測するにあたって，TaskPit に現在の Task の Aim を入力する機能を実装した．図 5 に機能追加後の TaskPit のスクリーンショットを示す．

提案する手法では Task の Aim が記録された作業履歴をトレーニングセットとして用いる．被験者が入力する Aim はそれぞれの被験者に対して事前にインタビューし，決定した．実験で利用された Aim を以下に示す．

- Research (Thesis)
- Research (Presentation)
- Research (Coding)
- Research (Survey)
- Research (Other)
- Homework (Signal Processing: SP)
- Homework (Software Engineering: SE)
- Homework (Other)
- Break
- Other

同様に記録される Task もそれぞれの被験者に対して行ったインタビューを元に設定した．図 6 に Task の設定例を示す．設定されていないアプリケーションが使用された場合には “not registered” として計測される．本実験では “not registered” を含む合計 23 種類の Task を計測した．

4.2 評価

提案手法の精度を被験者が入力した Aim と提案手法が予測した Aim を比較し評価する．Task の時系列を予測に

```
edit text = EmEditor.exe|TeraPad.exe|notepad.exe|WINWORD.EXE|
Acrobat.exe|acrodist.exe
edit presentation  = POWERPNT.EXE
data analysis = EXCEL.EXE
coding = VCExpress.exe|devenv.exe
debug = VCExpress.exe:Debugging|devenv.exe:Debugging
taskpit = TaskPit.exe|TaskAnalyzer.exe
web browse = iexplore.exe|chrome.exe|firefox.exe
mail = Outlook.exe|chrome.exe:gmail
file operation = explorer.exe
desktop = explorer.exe:Program Manager:unknown
network = FFFTP.exe|WinSCP.exe
chat = Skype.exe
painting = mspaint.exe
```

図 6 Task の設定例

用いることの有効性を評価するために，予測対象 Task の特徴のみを用いた場合と，予測対象の前後の Task の特徴を加えた場合の予測精度を比較する．本稿での実験では，被験者ごとに，前後の Task 考慮数を 1 から 100 まで変化させたときの予測精度を確認する．また将来的に，前のタスク情報のみを用いたリアルタイム予測への応用も考えられるため，前の Task のみ，後の Task のみを考慮したデータセットも作成し評価実験を行う．

表 1 に予測結果の例を示す．各列は被験者が入力した Aim を示し，各行は提案手法による予測結果を示す．列と行の項目名が一致する箇所の数値が各 Aim に対する正答数になる．本研究では予測精度の評価に実験データ全体の予測精度と Aim 毎の予測精度を求める．

実験データ全体に対する予測精度は以下の式で求められる．

$$All\_accuracy = \frac{\text{全ての Aim に対する正答数}}{\text{データセットの Task の総数}} \quad (2)$$

ある Aim A の予測精度は以下の式で求められる．

$$Aim\_accuracy = \frac{\text{A の正答数}}{\text{A を Aim とする Task の総数}} \quad (3)$$

評価実験では，two-fold-cross-validation を用いる．すなわりデータセットの半分を学習用に使用し，残りの半分を予測に使用する．また，各データセットに対し 10 回実験を行い，その結果の平均値を評価に使用する．

表 1 予測結果の例

		実際の Aim				
		Thesis	Coding	SW	SP	Break
予測された Aim	Thesis	140	11	7	0	35
	Coding	18	181	3	1	13
	SW	33	27	167	0	41
	SP	0	1	0	14	1
	Break	54	3	56	7	282

表 2 各被験者の Task と Aim の数

	Task 数	Aim 数
被験者 A	1130	8
被験者 B	637	5
被験者 C	1998	5
被験者 D	1550	6
被験者 E	1667	5
平均	1385.2	5.8
標準偏差	532.7	1.3

5. 結果と考察

5.1 全体の結果

図 7 に Aim と Task の計測例を示す．また，各被験者が記録した Task 数，Aim 数を表 2 に示す．

図 8 に各被験者の予測精度を示す．縦軸が予測精度，横軸が前後の Task の考慮数を示している．図から，前後の Task を考慮した場合の予測精度は，考慮しなかった場合に比べ大幅に向上していることがわかる．表 3 に異なる Task 考慮数ごとの予測精度の変化を示す．考慮数を 0 から 1 にすると平均で 14.2%精度が向上している．さらに考慮数を 10 まで増やすと精度がさらに向上し，平均で 88.2%と高い予測精度が確認できた．考慮数を 10 以上増やすと，被験者 A 以外の予測精度は向上し続け，4 人の被験者においては 90%以上の精度で予測することができた．以上の結果は，予測対象の Task について，前後の Task に関する情報が予測精度の向上に役立つことを示唆している．

5.2 Aim 毎の精度

提案手法では，実験期間中に Task 数が多かった Aim に対して，特に高い予測精度が見られた．表 4 に被験者 D の Aim と Task の数，それぞれの Aim に対する予測精度を示

```
Task Name,  Lclick, Rclick, Keystroke, Task Aim
edit text,       9,      2,         27, Not selected
edit text,      23,      9,        210, Homework(SE)
web browse,     14,      3,         53, Homework(SE)
another,         2,      0,          0, Homework(SE)
web browse,     12,      0,         19, Homework(SE)
data analysis,   9,      1,         22, Homework(SE)
edit text,       9,      0,        176, Homework(SE)
data analysis,  23,      2,         59, Research(Thesis)
edit text,       1,      0,         24, Research(Thesis)
```

図 7 Aim を記録した作業履歴例

表 3 Task 考慮数ごとの予測精度の変化

Task 考慮数	被験者 A	被験者 B	被験者 C	被験者 D	被験者 E	平均	標準偏差
0	69.6 %	57.7 %	54.4 %	55.8 %	51.4 %	57.8 %	0.0624
1	80.3 %	74.3 %	69.2 %	73.1 %	63.2 %	72.0 %	0.0568
10	88.3 %	92.8 %	87.0 %	90.6 %	82.4 %	88.2 %	0.0352
30	85.0 %	96.4 %	92.9 %	95.6 %	87.7 %	91.5 %	0.0446
50	84.8 %	96.5 %	93.6 %	96.5 %	90.1 %	92.3 %	0.0443
100	83.9 %	96.6 %	91.8 %	95.7 %	88.6 %	91.3 %	0.0467

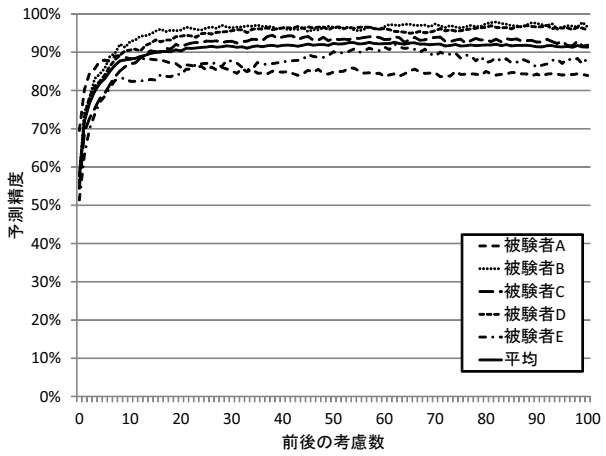

図 8 前後の考慮数と予測精度

す．予測精度は前後の Task を 10 個考慮した時のものを示す．被験者 D は実験期間中の Task 数が多かった Aim として *Research*(*coding*) と *Research*(*Thesis*) を行っており，それぞれに対する予測精度が 98.4%，96.1%と高い数値を示している．その一方で Task 数が少なかった *Break* と *Other* では予測精度は低かった．また，Task 数が少なくない *Homework*(*SE*) と *Homework*(*SP*) ではそれぞれ予測精度に差が出ていた．データセットを調査したところ，*Homework*(*SE*) では “not registered” の Task が作業量の多い特徴的なものであったが，連続した作業の中に頻繁に現れる Task ではなかった．前後の考慮数を増やすと *Homework*(*SE*) の予測精度が向上していたため，考慮数が少ない場合，Aim の主な Task に関する特徴が取得できず予測精度が低くなることが考えられる．反対に，高い予測精度が確認できた *Homework*(*SP*) では，この Aim 特有の Task(data analysis) があり，かつ作業量が多く頻繁に現れたことから予測精度が高かったと考えられる．このような傾向は多くの被験者でも見られた．この結果から，提案手法を用いることで，Task 数の多かった Aim に費やした時間の計測がより正確になるため，提案手法は PSP を用いる際に有用であると考えられる．

一方で被験者 A は，他の被験者と異なる結果を示しており，考慮数を増やすにつれて一部の Task の多い Aim で予測精度が減少した．表 5 に被験者 A の各 Aim に対する Task 数と割合を示す．被験者 A は実験期間中の主な Aim として *Research*(*coding*) と *Research*(*Thesis*), *Break* を行っており，それ以外の Aim は全体の Task に占める割合が 5%未満と低かった．図 9 に被験者 A の Aim 毎の予測精度を示す．

図からは *Research*(*coding*) と *Research*(*Thesis*) に対する予測精度が他の Aim に比べて特に高いことがわかる．特に前後の考慮数が 20 を超えると，この 2 つの Aim に対する精度は 100%に近い値が確認できた．一方で，Task 数が多い Aim である *Break* に対する予測精度が低下している．

被験者 A への実験後のインタビューと作業履歴から，被験者 A が *Break* を Aim としている Task のほとんどが “Web browse” であった．“Web browse” Task は被験者 A の場合，*Research*(*coding*) と *Research*(*Thesis*) にも多く含まれている．また予測結果から，*Break* を Aim にとる Task が *Research*(*coding*) や *Research*(*Thesis*) に誤って

表 4 前後 10Task を考慮したときの予測精度（被験者 D）

	Task 数	Task の割合	予測精度
Thesis	495	31.9 %	96.1 %
Coding	462	29.8 %	98.4 %
SE	228	14.7 %	73.6 %
SP	221	14.3 %	97.7 %
Break	140	9.0 %	63.7 %
Other	4	0.3 %	0.0 %

表 5 被験者 A の Task 数と割合

	Task 数	Task の割合
Research(Coding)	596	52.7 %
Research(Thesis)	261	23.1 %
Break	170	15.0 %
Homework(SP)	51	4.5 %
Homework(SE)	30	2.7 %
Research(Other)	16	1.4 %
Homework(Other)	1	0.1 %
Other	5	0.4 %

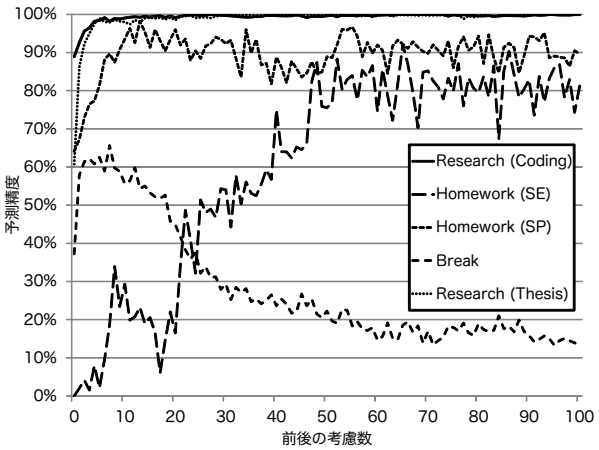

図 9 各 Aim の予測精度（被験者 A）

予測されている結果も確認することができた．

以上の点から提案手法では，似た Task の構成を持つ Aim が複数存在する場合，全体の予測精度が減少してしまうことが考えられる．

5.3 前のみ・後のみを用いた予測

前と後の Task の情報について何らかの差異があるかを確認するため，前のみ，後のみの Task 情報を用いた場合での予測精度について比較を行った．図 10 に各被験者の平均値を用いた比較結果を示す．図は縦軸が予測精度，横軸が考慮数を示している．

考慮数が少ない場合での予測精度では，前後と残り 2 つに差が見られる．これは予測に用いる特徴の数に差があるからだと考えられる．考慮数を増やしていくと 3 種類共に値が収束し，予測精度に差がないことが分かる．この結果は前と後の Task の情報の価値が同等であると示唆している．また前の Task の情報のみを用いた場合でも高精度の

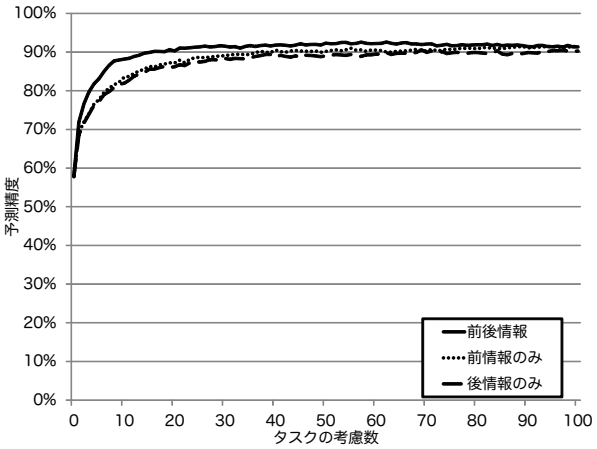

図 10 前のみと後のみを考慮した場合の予測精度

予測が期待できるため，提案手法を用いてリアルタイムに予測モデルを構築し，計測者に Aim を随時提示するシステムも考えられる．

5.4 特徴の重要度

Random Forests では，モデル構築時に使用した OOB データを用いて，使用している変数 (特徴) の重要度を推定することができる．具体的には，重要度を推定したい特徴の値をランダムに変更することで，次元数を保ったまま目的変数 (Aim) との関係性を無くし，どの程度 OOB 推定値が上昇したかで重要度を決定する．図 11，図 12 にそれぞれ前後の Task を 10 個，100 個考慮した場合の重要度上位 15 件の特徴を示す．縦軸が特徴名，横軸が重要度を示す．重要度は被験者 5 人の平均値であり，重要度の値が高ければ Aim の予測に役立つ特徴である．

両方の図から共通して見られるのは，$T.1stkey$ や $T.1stLclick$ などの作業量が多かった Task(主要な Task) の特徴の重要度が，前後に存在した Task の名前よりも高いことである．また，予測対象 Task のクリック数や打鍵数の重要度は他の特徴に比べて低いことが確認できた．この結果は他の考慮数においても同じ傾向が見られたため，本稿で提案した手法による予測精度の向上は，上記の特徴を用いて各 Aim 毎の主要な Task を学習できたからだと考えられる．

5.5 予測モデル構築時間

Task の考慮数を増やした時の予測モデル構築にかかる時間について，被験者 5 人の平均値を図 13 に示す．縦軸は時間 (秒)，横軸は Task の考慮数を表している．図から分かるように予測モデルの構築時間は，Task の考慮数を増やすにつれて特徴が増えるため線形的に上昇している．ただし，前後 100 個までの Task を考慮しても平均で約 100 秒，最も時間のかかった被験者でも約 180 秒であった．提案手法では過去の作業履歴から予測モデルを構築し予測す

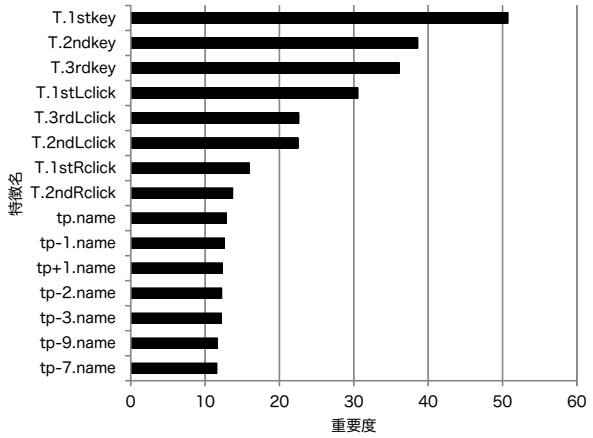

図 11 前後 10 個の Task を考慮した場合の重要度

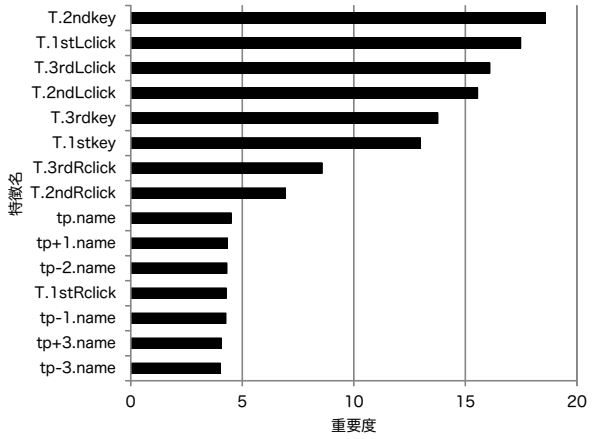

図 12 前後 100 個の Task を考慮した場合の重要度

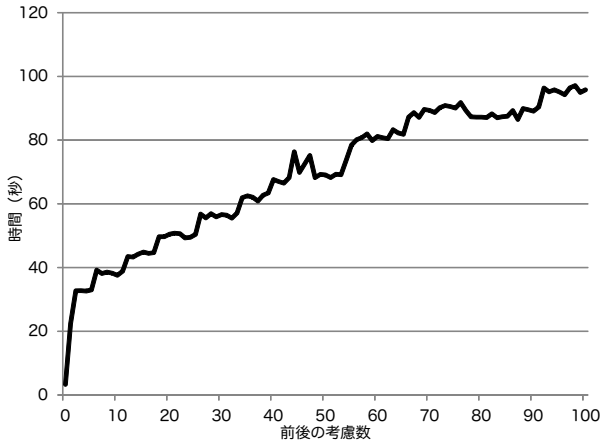

図 13 モデル構築時間

ることを考えているため，今回の実験結果からモデルの構築時間に関しては問題ないと考えられる．

6. おわりに

本稿では，コンピュータ上で行われる Task の Aim を予測する手法を提案した．提案手法は，PSP の支援システムである TaskPit によって記録された作業履歴から，予測対象の前後の Task 情報と機械学習アルゴリズムである Random Forests を用いて Aim を予測する．実験の結果は，予測対象である Task の前後に実行された Task の特徴が，予測精度を向上させる上で有用である事を示した．Task の考慮数を増加させると，85%から 95%近い精度を得ることが確認できた．特徴の重要度を調査した結果は，連続した Task の中で活発に行われている Task の識別が予測精度の改善に役立つ事を示唆している．また，実験期間中に記録された Aim のうち，Task が多かったものに対しての予測精度が高く，一部ではほぼ 100%の予測ができていた．よって，提案手法は精度の高い予測から Aim の入力を少なくし，ソフトウェア開発のプロセス計測を容易にすることが考えられる．

ただし，似た Task の構成を持つ Aim が存在する場合，予測精度の減少が見られた．そのため，新たに Aim を判別するための特徴が必要になる．例えば、作業ディレクトリを特徴として用いることで，Aim 毎の作業ディレクトリ，ファイルの違いから予測精度が向上する可能性がある．

また今後の展望としては，ソフトウェア開発現場でのデータ収集や実験を行い，実際にソフトウェア開発現場にて提案手法が有効であるかどうかを確認すること，Random Forests のパラメータを変更することによる予測精度の変化を確認することなどが挙げられる．前の Task 情報のみを用いたリアルタイム予測のシステム提案も含まれる．

参考文献

[1] W. S. Humphrey, “パーソナルソフトウェアプロセス入門,” 共立出版.
[2] 門田 暁人, 亀井 靖高, 上野 秀剛, 松本 健一, “プロセス改善のためのソフトウェア開発タスク計測システム,” ソフトウェア工学の基礎 XV, 日本ソフトウェア科学会 FOSE2008, pp.123-128, November 2008.
[3] M. Granitzer, A. S. Rath, M. Kroll, C. Seifert, D. Ipsmiller, D. Devaurs, N. Weber, and S. N. Lindstaedt, ”Machine Learning based Work Task Classification,” Journal of Digital Information Management, 2009, pp.306–313.
[4] S. Stumpf, X. Bao, A. Dragunov, T. G. Dietterich, J. Herlocker, K. Johnsrude, L. Li, and J. Shen, “Predicting User Tasks: I Know What You’re Doing!,” In Proc, the 20th National Conference on Artificial Intelligence (AAAI), 2005.
[5] N. Oliver, G. Smith, C. Thakkar, and A. C. Surendran, “SWISH: Semantic Analysis of Window Titles and Switching History,” In Proc. the 11th international conference on Intelligent user interfaces, 2006, pp.194–201.
[6] M. Bernstein, J. Shrager, and T. Winograd, “Taskpose: Exploring Fluid Boundaries in an Associative Window Visualization,” In Proc. the 21st Annual ACM Symposium on User Interface Software and Technology (UIST), 2008, pp.231–234.
[7] L. Breiman, “Random Forests,” Journal of Machine Learning, Vol.45, No.1, 2001, pp.5–32.