アジャイル開発プロジェクトにおける新人育成をサポートするタスク推薦システムの提案

黒木　春伸†,†　清　雄一†,‡　田原　康之†,‡　大須賀　昭彦†,‡

†電気通信大学　〒 182-8585　東京都調布市調布ヶ丘 1-5-1

E-mail: †kurogi2756kenkyu@gmail.com, ‡{seiuny, tahara, ohsuga}@uec.ac.jp

あらまし　本論文では，アジャイル開発プロジェクトにおける新人育成をサポートすることを目的に，トピックモデリングの手法である LDA を用いたタスク推薦システムを提案する．近年，ソフトウェア開発においてアジャイル開発を採用するプロジェクトが増加傾向にあるため，アジャイル開発に特化した新人育成が重要となってきている．従って，アジャイル開発初心者に対して理解しやすいタスクを推薦することで，新人の不安や不満の解消に加えて，そのメンターの仕事量の軽減に繋がる．本研究では，アジャイル開発の中でもスクラム開発に焦点を当て，スプリントバックログのタスクデータセットに LDA を適用しモデルを作成した．次に，推薦したいタスクと新人のプロフィールにそれぞれモデルを適用しトピック値を得ることで，タスクと各新人のトピック値の類似度から各新人に適したタスクを推薦した．結果として，３位以内の推薦率の平均値は 0.803 となり，メンターを支援するタスク推薦システムとしては有効であることが分かった．

キーワード　ソフトウェア工学，アジャイル開発，トピックモデリング，LDA，推薦システム，新人育成

1. はじめに

ソフトウェアやシステムを開発する企業（SIer など）は，顧客に求められているシステム・ソフトウェアを正確に把握し，必ず期日までにリリース（納品）することが求められている．従って，不適合品の納品や納期遅れが生じてしまうと顧客の信頼を失うことに繋がる可能性があるため，それらの発生を防ぐことがとても重要である．そのためには，与えられた仕事を適切にこなす人材が必要不可欠であるため，優秀な人材の確保や育成を重要視している企業は多く[1]，新人の育成に関しては様々なモデルが提案されている[2]．そのモデルの中でも昨今では，新人と経験豊富な人（メンター）を繋ぐメンタリング[3]や，実際の開発プロジェクトに入り仕事をしながら学ぶ OJT[4]などがよく行われている．特にこのメンタリングでは，新人の活動を活発化（積極的な質問など）させる[5]．しかし，このメンタリングでは，エンジニア上級者を新人のメンターとする場合が多く，その場合，プロジェクトの円滑な進行に必要不可欠な資源を奪うことに繋がってしまう可能性がある[6]．また，メンターが忙しい場合，新人の疑問の解消に献身的になることが難しくなってしまう．従って，ほとんどの開発チームは，新人が自分でソースコードを探索するなど，疑問を自分で理解し解決することに期待している[7]．しかし，それによって新人の疑問を満足解決できない場合などは，新人の不満や不安の増大に繋がってしまう[8]．

そこで，新人にこなしやすく理解しやすいようなタスクを割り当てることができれば，エンジニア上級者の忙しさに関係なく新人自身の疑問解決に期待することができ，プロジェクトを円滑に進めながら効率良く新人育成が行えると考えた．

さらに，近年では，ソフトウェア・システム開発において様々な開発手法が存在するが，その中でもアジャイル開発プロジェクトが増加傾向にある [9] [10]．従って，その分他開発手法より，新人育成の制度化や効率化が重要である．また，アジャイル開発のタスク割り当ての研究において，プロジェクトマネージャーやプロジェクトの成功を目的としたタスク推薦システムはいくつか提案されている．しかし，新人育成を目的としたタスク推薦システムの研究は少ない．また，アジャイル開発ではサイクルごとにタスクの内容が変わることが多いため，アジャイル開発やプロジェクトに対しての理解が浅い新人の不安や戸惑いは大きいと考えられる．従って，アジャイル開発初心者に対して，タスク推薦システムが有効であると考えた．

本研究では，アジャイル開発の中でもスクラム開発に焦点を当てており，約 2500 からなるアジャイル開発タスクデータセットにトピックモデルを適用している．そして，新人データとタスクデータのトピック類似度を比較することで，タスクを推薦するシステムを構築した．

本稿の構成は以下の通りである．2 章では，アジャイル開発について，また，様々な開発手法がある中でアジャイル開発を選んだ理由について説明する．3 章では，関連研究を述べる．4 章では，本研究で使用したクラスタリング手法であるトピックモデル（LDA）について説明する．5 章では，提案手法について述べ，6 章で，実験内容を述べる．7 章では，結果と評価について述べ，8 章でまとめを述べる．

2. アジャイル開発

本章では，本研究で焦点を当てたアジャイル開発とスクラム開発について説明する．

まず，アジャイル開発とは，ソフトウェア開発に用いられる開発手法の一つであり，要件定義，設計，開発，テスト，リリースという開発工程を短いスパンで繰り返す手法である．アジャイル開発のメリットとして，トラブル発生時や顧客のニーズに応じて，修正や変更が楽であることや，開発スピードが速いことが挙げられる．デメリットとしては，プロジェクト全体のスケジュール管理の難しさや，少しずつ開発を進めていくため，どのような完成品になるかが想像しにくいといったものがある．

次に，スクラム開発について説明する．スクラム開発とはアジャイル開発のフレームワークとしてよく利用される手法の一つである．スクラム開発の主なプロセスを簡単に示す．（図 2.1）

まず，スプリントとは，ある量のタスクを達成する際に設ける短く区切られた期間のことである．一般的に 1 スプリント 1~4 週間ほどで構成される．

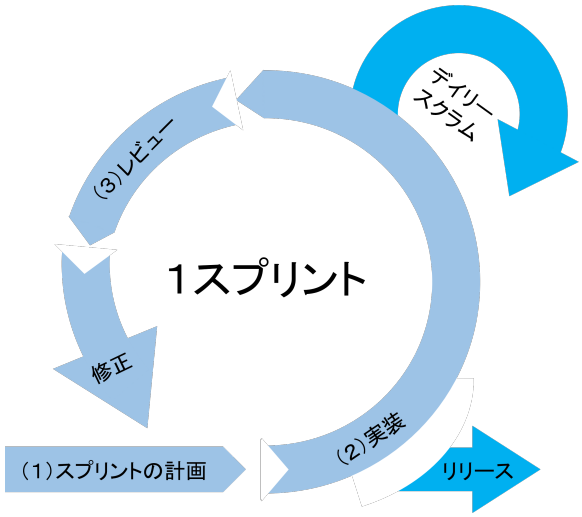

図 2.1　スプリントの例

（1）　スプリントの計画

1 サイクル期間中に完成させるタスクをリストアップしたものを作成する．これをスプリントバックログといい，利用者（エンドユーザ）視点のソフトウェアの機能，タスク名，タスクの説明，タスクの優先度，残存タスク，各タスクの完了時間，1 日のタスクに対する時間割り当てなどが含まれる．

（2）　実装，デイリースクラム

実装しながら，毎日数十分程度のミーティングを行い，進捗状況の確認や割り当てられていたタスクが適切かを話しあう．

（3）　スプリントレビュー

スプリントで開発されたソフトウェアの振り返りを行い，次のスプリントに備える．

このように，上記の（1）~（3）を繰り返すことによってよりよい製品の開発を目指すのがスクラム開発である．

本研究では，このスクラム開発のスプリントバックログに着目し，新人に対しスプリントバックログのタスクの割り当て時，最も新人に適しているタスクを推薦してくれるような推薦システムの構築を目的としている．

3. 関連研究

3.1 RE 活動を支援する推薦システム

まず，RE とは要求工学のことであり，ソフトウェア開発におけるユーザー要求を仕様化するプロセスおよび研究のことをいう．この RE 活動を支援する推薦システムの研究としては，ステークホルダーの特定，要求の引き出し，要求の優先順位付け，リリース計画など，様々な推薦システムが提案されている．例えば，Shambour らは，RE 活動を支援するための HCBCF アプローチ [11]を提案した．これは，ソフトウェア要件リポジトリから関連する要件を特定することで，プロジェクト関係者が要件を欠落するリスクを軽減することを目的としている．

しかし，このような RE 活動を支援するタスク推薦システムは，コーディングタスクのことは考えられておらず，RE 活動からコーディングまでを短期間で行うアジャイル開発には不向きである．

3.2 新人のコーディングタスクを支援する推薦システム

Raul Medeiros らが提案した RecomMentor [12]は，SPL 開発においてメンターをサポートするタスク推薦システムである．

まず，SPL 開発とは，過去に開発した類似製品の一部分を再利用して，新製品を開発することである．これにより，コストの削減や市場投入までの大幅な削減が見込まれる．

RecomMentor の仕組みとしては，まず，SPL プロジェクトのコードベース（ソフトウェア・システムなどを構築するために使用されるソースコードの集まり）を用いてトピックモデルを用いて分析する（トピックモデルについては次章で説明する）．次に，新人のプロファイルデータに対しても同様にトピックモデルを適用する．最後に，コードベースと新人プロファイルデータのトピック類似度を比較し，高い順に新人にタスクを推薦するという仕組みである．また，

RecomMentor では，SPL のコードベースと新人プロファイルの取得に WACline [13]を使用し，トピックモデルを用いて分析を行う LASCAD [14]上に構築されている．

本研究では，トピックモデルは採用しているが，アジャイル開発に焦点を当て，コーディングタスクのような下流工程以外の要件定義や設計などの上流工程のタスクも推薦が可能である．

3.3 アジャイル開発を支援する推薦システム

アジャイルソフトウェア開発におけるタスク割り当てについては様々な研究がされている．例えば，Aslam ら [15]はチームメンバーの技術的な好み，専門知識，および現在の作業量に基づいて，効率的なタスク割り当てを実現する理論的な枠組みを提案した．次に，Saad Shafiq らは，プロジェクトマネージャーを支援するためのタスク推薦システムである TaskAllocator [16]を提案した．TaskAllocator は，過去に割り当てられたタスクのテキストから特徴量を LSTM で学習し，タスクにあった役割（フロント・バックエンド，デベロッパーなど）を推測するものである．従って，TaskAllocator は PM を支援するために各タスクに最適な役割を推薦するのに対し，本研究ではメンターを支援するために新人に最適なタスクを推薦する点が違いとなっている．

4. トピックモデル

4.1 統計的潜在意味解析

トピックモデルの簡単な例を図 4.1 に示す．トピックモデルとは統計的潜在意味解析の一つであり，データ（文章など）に潜む「意味」を統計的に解析する手法である．図 4.2 を見ると，「就職活動」「大学」「物理」に関する単語が集まっていることが分かり，このように，複数の単語の共起性によって創発されるものを「潜在的意味」と考える [17]．また，それぞれの単語の集まりは潜在的意味のカテゴリと考えることができ，これを「潜在トピック」または「トピック」という．従って，図 4.2 の各カテゴリは「就職活動トピック」「大学トピック」「物理トピック」と解釈できる．

さらに，図 4.2 を詳しく見ると，大学トピックと物理トピックで「単位」という単語が含まれている．これは，「単位」の意味カテゴリが複数解析したデータ（文章）には存在するということを表している．

従って，トピックモデルでは，トピックを用いて各文書をトピックの集合として表現することで，各文書を分類することが可能となる．

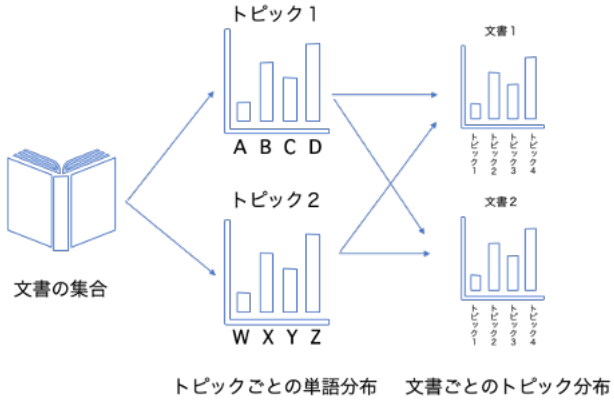

図 4.1　トピックモデリングの簡単な例

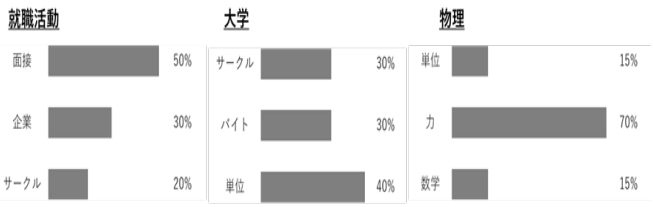

図 4.2　単語の集まりの例

4.2 LDA

本節では，統計的潜在意味解析を可能にする統計モデルである，Latent Dirichlet Allocation(LDA)について簡単に説明する．

LDA では，ディリクレ分布で文書のトピック割合と各トピックの単語選択確率を生成し，そのトピック割合をパラメータとした多項分布からトピックごとの単語数を生成する．次に，ディリクレ分布と多項分布で生成した各トピックの単語選択確率と単語数から各単語の使用回数を生成している．

トピックに対する語の生成確率は$p(w^j = 1 \mid z^i = 1) = \beta_{i,j}$（$w_n$は各 n 個の単語，$z_n$は多項分布からサンプリングされた 1 つのトピック）である行列$\boldsymbol{\beta}$によって表される．文書における単語の出現分布を考えると，単語の分布は偏っているはずであり，それをパラメータ$\boldsymbol{\alpha}$によって表現することを可能としている．　学習データとテストデータからパラメータ$\boldsymbol{\alpha}$と$\boldsymbol{\beta}$が与えられたとき，トピック混合$\boldsymbol{\theta}$の同時分布と N 個のトピック $\mathbf{z}$ の集合と単語の集合 $\mathbf{w}$ の集合は（1）式となる．

$$p(\theta, z, w|\alpha, \beta) = p(\theta|\alpha)\prod_{n=1}^{N} p(z_n|\theta)p(w_n|z_n, \beta) \qquad (1)$$

$\boldsymbol{\theta}$ に関して積分し，$\mathbf{z}$ に関して和を求めることで文書の生成確率を得られる（2 式）．[18]これにより文書をトピック確率分布によって表すことができる．

$$p(w|\alpha,\beta)=\int p(\theta,\alpha)\left(\prod_{n=1}^{N}\sum_{z_n}p(z_n|\theta)p(x_n|z_n,\beta)\right)d\theta \quad (2)$$

表 4.1 は，文書「タスク１」のトピックに関する表現例，表 4.2 は「トピック 0」の単語での表現例を示す．

表 4.1　文書「タスク１」のトピックによる表現例

トピック１	トピック２	トピック３	トピック４
0.024779	0.549779	0.024073	0.401367

表 4.2　「トピック０」の単語表現例

layer	game	business	system	…
0.011	0.01	0.01	0.01	…

4.3 LDA 評価手法

LDA を用いて作成したモデルの評価手法として，Perplexity と Coherence がある．

まず, Perplexity とは分岐数または選択肢の数を表している．例えば，ある文書の中から 1 単語を探し出すとき，文書の語彙数が 1000 個であれば，ランダムなモデルでは単語の選択肢が 1000 個あることになる．この時，LDA による Perplexity が 100 であれば，探し出す 1 単語の選択肢の数を 100 まで絞ることが可能であることを意味する．このように，Perplexity が低いほど良いモデルと考える．

ここで Perplexity（PPL）の計算式を(3)式に示す．

$$PPL=\exp\left(-\frac{1}{N}\sum_{i=1}^{N}\log_2 p(w_i|\theta)\right) \quad (3)$$

この時，N を出現単語数，$p(w_i|\theta)$を周辺の単語に対する単語w_iの発生確率とする．

計算方法としては，まず各文書の単語を学習データとテストデータに分割する．配分としては，9:1，8:2，5:5 などがよく用いられる．そしてモデルに従って，学習データでパラメータ θ を求め，テストデータを用いて$p(w_i|\theta)$が最大になるようにパラメータを更新していく．

この Perplexity は，主に汎用能力を測定する指標である．その他に，抽出されたトピック中の単語の性質に関する評価も行われており, Perplexity と共にトピックモデルの評価手法として広く使われているのが，Newman らによる Coherence [19]という評価手法である．

まず，トピック k において，出現確率の高い順に c 個の単語を選び，$W_k=\{w_1,w_2,\dots,w_c\}$とする．この時，単語間のある類似度関数$sim(w_i,w_j)$に対して，$W_k$の Coherence を(4)式に示す．

$$Coherence_{sim}(W_k)=\frac{2}{c(c-1)}\sum_{1\leqq i\leqq c-1}\ \sum_{j+1\leqq j\leqq c}sim(w_i,w_j) \quad (4)$$

（4）式から，Coherence は単語間類似度の平均であり，W_k内の単語間の類似度が高いほど Coherence は大きくなるため，トピック全体の Coherence が高ければ良いモデルであるといえる [17].

従って，Perplexity ができるだけ低く，Coherence ができるだけ高くなるようにモデルを調整し，適切なトピック数を決定することが重要である．

5. 提案手法

5.1 概要

図 5.1 に提案手法のタスク推薦システムの構成図を示す．

簡単に提案手法の概要を以下に示す．

① 訓練データとテストデータを用意
② 各種データに前処理を実行
③ 訓練データからモデルを作成
④ モデルをテストデータに適用し，タスクデータと新人データそれぞれのトピック値を算出
⑤ トピック値の類似度を比較し，推薦順位決定
⑥ タスクを推薦

以上の流れで新人に適切なタスクを推薦する．

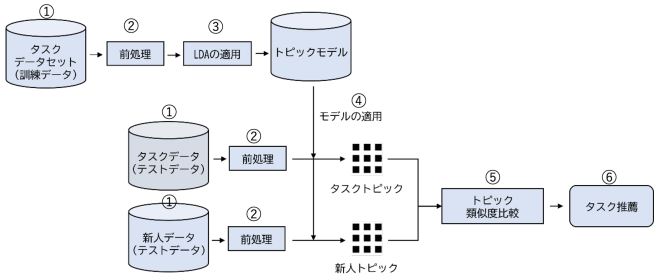

図 5.1　タスク推薦システムの構成図

本研究では，実用性を考え，タスクトピックと新人トピックを算出する際，ランダム性がないように設定している．従って，同じ訓練データとテストデータを用いた場合，常に同じ結果が出力される．

5.2 データセット

データセットは主に Taiga.io[20]にて収集を行った．Taiga.io とは，アジャイル開発に特化したプロジェクト管理プラットフォームであり，タスク管理方法としてかんばん方式とスクラム方式を選択することができ

る．また，公開プロジェクトが 23 万個以上あるため，様々なスクラム開発のデータを収集できる．

以下にプロジェクトの選択基準を示す．

（1） タスク，ユーザーストーリー，課題が合わせて 100 個以上であること

（2） 少なくとも 5 人のチームメンバーで構成されていること

（3） 少なくとも 5 つのスプリントがあること

上記の選択基準で収集し，合計約 3000 のタスク文章を収集した．

次に，このデータセットに対して行った前処理を以下に示す．

（1） URL の削除

（2） 改行コードの削除

（3） 形態素解析

（4） 大文字を小文字に変換

（5） ストップワードの削除

“in”, “is”, “are”, “the” などの英語のストップワードは必要ではないため削除した．

（6） 特定の単語の削除と統合

例えば，”friend”, “please” など，明らかに必要ではない単語を削除した．また，”frontend” と “front” は同じ意味合いで使用されるため，これらを ”front” に統合した．

（7） 辞書の作成

（6）までで作成された単語の集合に対し，単語の出現が x 文書に満たない単語と，y%以上の文書に出現する単語を極端とみなし削除した．本研究では，x = 5, y = 10 で行った．

以上の手順により，データセットを LDA のモデルに適用できるようにした．その結果，特徴ありとして認識された単語数が 2168 個，文書数が 2508 文書となった．

5.3 モデルの作成，評価，適用

前節で述べたように，データセットから作成された辞書をもとに LDA を適用し，モデルを作成する．また，この時，モデル作成に最適なトピック数を決定する必要性があるため，モデルの評価指標として一般的に用いられる Perplexity と Röder らによる Coherence [21]を用いる．本研究では gensim のライブラリを用いて，モデルの作成から評価まで行っている．

次に，このモデルを適用するための新人プロフィールデータとタスクデータを用意する．このデータに対してもデータセットと同様に前節 5.2 の（1）～（6）と同じように前処理を実行する．そして，それらのデータに対して，作成したモデルを適用し，各新人と各タスクのトピック値を算出する．

最後に，各新人のプロファイルデータと各タスクデータのトピック値を比較し，各新人に対してトピック類似度の高い順にタスクを推薦する．

5.4 トピック類似度の判定手法

トピック類似度の比較に関しては，コサイン類似度を用いて比較する．5.3 節にて各新人と各タスクにおいて，それぞれのベクトルが算出される．ここで，m をトピックの数として，新人 i のベクトルを $\overrightarrow{N_i} = [N_1, N_2, \dots, N_m]$, タスク j のベクトルを $\overrightarrow{T_j} = [T_1, T_2, \dots, T_m]$, とするとコサイン類似度 $CosSim_{ij}$ は(5)式のように表される．

$$CosSim_{ij} = \frac{\overrightarrow{N_i} \cdot \overrightarrow{T_j}}{\|\overrightarrow{N_i}\| \|\overrightarrow{T_j}\|} = \frac{\sum_{k=1}^{m} N_{ik} T_{jk}}{\sqrt{\sum_{k=1}^{m} N_{ik}^2} \sqrt{\sum_{k=1}^{m} T_{jk}^2}} \quad (5)$$

コサイン類似度では，値が 1 に近づくほど類似度が高く，-1 に近づくほど類似度が低いいことを表すため，この類似度の値が高い順に新人にタスクを推薦する．

6. 実験

6.1 実験用データ

まず，前章 5.2 の前処理(6)の x と y を決定する．本研究では，データセット数が少ないため，x=1, y=40 とすることで，できるだけ単語数を増やした．

次に，トピック数を決定するために，Perplexity と Coherence を求めた．Perplexity と Coherence を求めるにあたって，トピック数を 1 から 21 まで変化させ，どこで Perplexity が低く，Coherence が高くなるかを測定した（図 6.1）．その結果，トピック数が 4 と 12 の時，Coherence が周りのトピックより高くなっていることが分かる．そして，その 2 つの Perplexity を見ると，約 2 倍の差が生じていることがわかる．また，データセット数が少なく単語数も少ないことも考慮し，トピック数ができるだけ少ない方がよいと考え，トピック数は 4 が最適だと判断した．

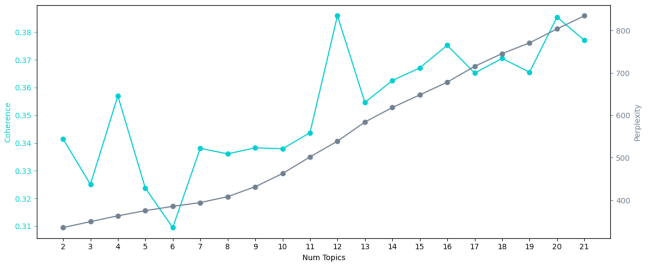

図 6.1 Perplexity と Coherence

6.2 実験用データ

実験用に新人プロファイルデータとタスクデータを用意した.

まず，新人プロファイルデータに関しては，新規卒業者（以降，新卒）が 6 人，中途採用者（以降，中途）が 5 人の 11 人分用意した（表 6.1）.

新卒 1~6 は，電気通信大学のホームページ[22]の学科説明文を使用した．中途 1~5 は，NTT データ[23]とアクセンチュア[24]の中途採用募集要項の文章を使用した．デザイナーに関しては募集内容の多かったアクセンチュアを採用している.

表 6.1　新人のプロファイルデータ

	専攻・前職	情報参照元
新卒１	メディア情報学	大学ホームページ
新卒２	経営・社会情報学	大学ホームページ
新卒３	情報数理工学	大学ホームページ
新卒４	コンピュータサイエンス	大学ホームページ
新卒５	セキュリティ情報学	大学ホームページ
新卒６	電子情報学	大学ホームページ
中途１	システムエンジニア	NTTデータ
中途２	データサイエンティスト	NTTデータ
中途３	セキュリティ専門	NTTデータ
中途４	デザイナー	アクセンチュア
中途５	プロジェクトマネージャー	NTTデータ

次に，タスクデータに関しては，データセットと同様に，Taiga.io を用いて，同じ選択基準で，６つ収集した（表 6.2）.

タスクデータとして，システム開発系が４つ，ゲーム開発系が 2 つ用意し，できるだけ違う種類のプロジェクトを採用した.

表 6.2　タスクデータ

プロジェクト名	プロジェクトの区分
App de ticket	チケット購入アプリ
reroils	図書館のシステム
Point of Service	統合システム
OS-WASABI	警告システム
Playtherapy	ビデオゲーム
Cobalt Team	ゲームアプリ

そして,新人プロファイルデータとタスクデータ共に,5.2 の（1）~（6）と同じように前処理を実行する.

最後に､新人プロファイルデータとタスクデータ共に作成したモデルを適用し、それぞれトピック値を算出した。そして、コサイン類似度を用いて、トピック類似度の高い順から新人にタスクを推薦した。

6.3 評価手法

LDA を用いて出力した結果に対する評価として，あらかじめ各新人に対して各タスクがどれだけの類似度を持っているかを示した表（表 6.3）を用意し，その表と出力結果がどのぐらい一致しているかで評価する．評価シートに関しては，著者等で作成した.

表 6.3　類似度評価シートの例

	プロジェクト１								
	task1	task2	task3	task4	task5	task6	task7	task8	task9
新卒１	○								○
新卒２				○	○				
新卒３				○	○				
新卒４				○	○				
新卒５				○	○				
新卒６				○	○				
中途１				○	○				
中途２									○
中途３				○	○				
中途４	○		○						○
中途５		○	○						

○の基準としては，新人とタスクの類似度を 1（近）~5（遠）で判定し，１のみ○をつけた．また，ここでは１新人に対して○のつくタスクは最大で３つとしている.

評価基準としては，推薦順位 1 位のタスクはどれだけ評価シートと一致しているか．また，推薦順位 3 位以内に○のついたタスクが推薦されているかによって評価する.

6.4 ベースラインモデル

本研究では，モデル作成時アルゴリズムにランダム性を用いたものをベースラインモデルとした．従って，このモデルは出力結果がランダムとなるため，モデル作成からトピック値の比較を 15 回実行し，各推薦順位に応じて重みを付けることで，出来るだけ一定の出力を行えるようにした．ここで，rank を順位とすると，順位の２乗の合計値 sum_i（i はタスク番号）が(8)式のように表される.

$$sum_i = \sum_{k=1}^{15} rank^2 \tag{8}$$

この時，rank を２乗しているのは，順位が低いほど重みを大きくしたいからである．従って，sum_i が低い順にタスクを各新人に推薦する．

7. 結果と評価

7.1 トピック値算出結果と考察

表 7.1 より，トピックごとに分類されていることが分かる．まず，topic0 では，”issue”, “fix”, “test”, “ui”などの単語が選ばれており，ui 系などの問題を解決やデバッグなどに関するトピックであると考えられる．Topic1 では，”update”, “test”, “server”, “system”などの単語が選ばれており，システムの更新についてのトピックであると考えらえる．次に，topic2 では，”layer”, “test”, “game”, “ui”などの単語が選ばれており，ゲームに関するトピックであると考えられる．最後に topic3 では，”business”, “service”, “appointment”などの単語が選ばれており，コーディングなどの開発系の単語ではなく，ビジネス関係に関するトピックであると考えられる．

しかし，４つのトピック中３つに”ui”が含まれているなど，同じ単語が複数のトピックに表れており，その分トピックの意味付けが難しくなっているとも言える.その原因としては,図 5.1 の Perplexity と Coherence の値からトピック数を決めたが，データセットが少ないため Coherence が全体的に低いことが考えられる．

表 7.1　データセットの各トピック算出結果

topic0	topic1	topic2	topic3
issue	update	layer	page
file	test	test	business
fix	page	game	service
test	server	report	time
page	file	business	appointment
ui	service	ui	webinar
front	ui	functionality	image
update	species	code	code
layer	screen	front	update
code	business	camera	list
screen	system	api	screen
developer	list	admin	movement
case	player	service	web
api	api	group	get
server	design	form	station

加えて，データセットに大きな偏りが生じていたことも，原因として考えられる．従って，適切な分類を行うためには，偏りがないようにデータセットを整えることや，データセットを 10 倍以上にすることで単語量を大幅に増やすことが必要だと考えられる．

7.2 タスク推薦の評価結果と考察

表 7.2 より，３位以内の推薦率を見ると 8 割を超えており，３位以内に新人に適切なタスクを推薦することが可能である確率が高いことが分かった．また，ベースラインモデルと比較しても 1 位推薦率，2,3 以内推薦率の全てが高くなっていることが分かる．

しかし，１位推薦率の平均値は 0.409 となり，５割を下回る結果となってしまった．この原因としては，タスクデータと新人データの単語数が少なく，モデル作成時に用いた単語の中に含まれていない単語テストデータに存在しており，モデル内に存在する単語のみで類似度が判定されるためである考えられる．

加えて，新人データの６人の新卒データに関しては，電気通信大学ホームページに記載されていることをデータとして扱っており，学科による差があまり生じていない可能性がある．従って，その結果，各新卒でトピック値に変化が生じておらず，その分１位に推薦されにくくなっていることも考えられる．

また，新人データとタスクデータの文章量が少なく，文書によって大きな違いが生まれなかった可能性も考えられる．

表 7.2　タスク推薦の評価結果

プロジェクト名	1 位 推薦数	1 位 推薦率	2 位以内 推薦数	2 位以内 推薦率	3 位以内 推薦数	3 位以内 推薦率
App de ticket	4	0.364	8	0.727	10	0.909
Playtherapy	7	0.636	9	0.818	9	0.818
reroils	5	0.455	8	0.727	10	0.909
OS-WASABI	3	0.273	6	0.545	9	0.818
Point of Service	2	0.182	5	0.455	7	0.636
Cobalt Team	6	0.545	6	0.545	8	0.727
平均	4.5	0.409	7	0.636	8.833	0.803
ベースラインモデルの平均	3.167	0.288	6	0.546	8.167	0.742

しかしながら，このシステムの目的はあくまでメンターを支援することである．そのため，１位ではなくても３位以内に適切なタスクが推薦されていれば，その３つのタスクの中から最適なタスクをメンターが選ぶことでき，効率の良い新人育成には繋がると考えている．

7.3 評価の妥当性

評価シートについては，アジャイルプロジェクトの実務経験が数年あるような人物が作成したものではないため，この部分によって研究結果が違うものになっている可能性は十分に考えられる．

8. まとめ

本研究では,近年のアジャイル開発プロジェクトの増加に対し，アジャイル開発に特化した新人育成が益々重要になっていくと考え，その新人育成をサポートできるようなタスク推薦システムを構築した．そのシステム構築にあたり，LDA を用いて新人とタスクのトピック値を算出し，それらを比較することでタスクを推薦できるようにした．この推薦システムの課題としては，構築したトピックモデルがデータセットの少なさから，テストデータに対して不適であることが挙げられる．従って，このタスク推薦システムを実用レベルにするには，データセットの大幅な拡張や，文書を比較する手法として LDA 以外の手法を取り入れる必要性があると考えている．

謝　辞

本研究は JSPS 科研費 JP21H03496, JP22K12157 の助成を受けたものです．

参　考　文　献

[1] 城山憲明，“企業における人材育成のあるべき姿，” *商大ビジネスレビュー*，第 5 巻，第 4 号，pp. 43-54, 2016.

[2] G. G. Sharma and K. J. Stols, "Exploring onboarding success, organizational fit, and urnover intention of software professionals.," Journal of Systems and Software, no. 159, 2020.

[3] J. D. Kammeyer-Mueller and T. A. Judge, "A quantitative review of mentoring research: Test of a model," Journal of Vocational Behavior, vol. 72, no. 3, pp. 269-283, 2008.

[4] W. J. Rothwell and H. C. Kazanas, "Planned OJT is productive OJT," Association for Talent Development (ATD), vol. 44, no. 10, 1990.

[5] F. Fagerholm, A. S. Guinea, J. Münch and J. Borenstein, "The role of mentoring and roject characteristics for onboarding in open source oftware projects," ESEM, pp. 1-10, 2014.

[6] R. Pham, S. Kiesling, L. Singer and K. Schneider, "Onboarding inexperienced developers: struggles and perceptions regarding automated testing," Software Qual. J., vol. 25, no. 4, p. 1239–1268, 2016.

[7] G. Viviani and G. C. Murphy, "Reflections on onboarding practices in mid-sized companies," IEEE/ACM 12th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE), pp. 83-84, 2019.

[8] A. Ju, H. Sajnani, S. Kelly and K. Herzig, "A Case Study of Onboarding in Software Teams: Tasks and Strategies," ICSE, pp. 613-623, 2021.

[9] L. Balboni, "CollabNet VersionOne Announces the 12th Annual State of Agile Report," CollabNet, 2018.

[10] R. Hoda, N. Salleh and J. Grundy, "The Rise and Evolution of Agile Software Development," IEEE Software, pp. 2-7, 2018.

[11] A. H. H. Qusai Y. Shambour*, Q. M. Kharma and M. M. Abualhaj, "Effective Hybrid Content-Based Collaborative Filtering Approach for Requirements Engineering," Computer Systems Science and Engineering, vol. 40, no. 1, pp. 113-125, 2021.

[12] R. Medeiros and O. Dıaz, "Assisting Mentors in Selecting Newcomers’ Next Task in Software Product Lines.:A Recommender System Approach," CAiSE, pp. 460-476, 2022.

[13] H. Medina, O. Diaz and X. Garmendia, "WACline: A Software Product Line to harness heterogeneity in Web Annotation," SoftwareX, vol. 18, 2022.

[14] D. Altarawy, H. Shahin, A. Mohammed and N. Menga, "Lascad : Language-agnostic software categorization and similar application detection," Journal of Systems and Software, vol. 142, pp. 21-34, 2018.

[15] W. Aslam and F. Ijaz, "A Quantitative Framework for Task Allocation in Distributed Agile Software Development," IEEE Access, vol. 6, no. 15, pp. 380-390, 2018.

[16] S. Shafiq, A. Mashkoor, C. Mayr-Dorn and A. Egyed, "TaskAllocator: A Recommendation Approach for Role-based Tasks Allocation in Agile Software Development," International Conference on Global Software Engineering (ICGSE), 2021.

[17] 佐藤一誠，トピックモデルによる統計的潜在意味解析，コロナ社, 2015.

[18] 高．輝彦，高．正則 ，勅．可海，“LDA を用いた類似項目検索のための前処理法，” 情報処理学会シンポジウム論文集，第 2 巻 2013, pp. 85-92, 2013.

[19] D. Newman, J. H. Lau, K. Grieser and T. Baldwin, "Automatic Evaluation of Topic Coherence," Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pp. 100-108, 2010.

[20] Kaleidos, “Taiga: Your opensource agile project management software,” https://tree.taiga.io/discover.

[21] M. Röder, A. Both and A. Hinneburg, "Exploring the Space of Topic Coherence Measures," Proceedings of the Eighth ACM International Conference on Web Search and Data Mining, pp. 399-408, 2015.

[22] 電気通信大学，“学域（学部）・大学院｜” https://www.uec.ac.jp/education/.

[23] 株式会社 NTT データ，“募集職種一覧|UpToData ／NTT データの人と仕事、キャリアを伝える WEB マガジン（経験者採用）用），”．https://www.nttdata.com/jp/ja/recruit/careers/recruit/

[24] アクセンチュア，“Search Jobs | Accenture,” https://www.accenture.com/jp-ja/careers/jobsearch?jk=&sb=1&vw=0&is_rj=0&pg=1.