目的と手段を仲介するクエリ推薦手法の提案

猪口　真吾† 　奥　 健太†

† 龍谷大学大学院 理工学研究科　〒 520-2194 滋賀県大津市瀬田大江町横谷 1-5
E-mail: †t19m056@mail.ryukoku.ac.jp, ††okukenta@rins.ryukoku.ac.jp

あらまし　プログラミング初学者にとって，あるプログラミング言語である課題を解決する際に適切なクエリを入力することは困難である．本研究では，プログラミング初学者に対して課題解決に関連するクエリを推薦するシステムを提案する．提案システムでは，ユーザが目的と手段に関するクエリを一つずつ入力すると，その手段で目的を達成するためのより詳細なクエリが推薦される．ユーザはこれら推薦クエリを目的クエリとして用いることで再検索する．この過程を繰り返すことで，ユーザは課題解決に向けて学習すべき項目を絞り込んでいくことができる．このようなシステムを実現するためには，クエリ間の関連性に基づき，目的クエリと手段クエリを仲介するクエリを探索することが課題となる．本研究では，技術共有サービスである Qiita(注3) を基にクエリ間の関連性を算出する．

キーワード　クエリ推薦

1. はじめに

プログラミングにおいて，開発と並行して，Google(注1)や Yahoo! Japan(注2)等の Web 検索を用いて課題解決法を探索することはよくやられている．多くの場合，1 つの課題を解決するために複数回の Web 検索を行い，多数のサイトを渡って解決法を見つけ出す．この行動は，多数の情報に触れることで未知の技術を知る機会になる．しかし，プログラミング初学者は，技術やツールに関する知識が乏しいため検索クエリの選定が難しく，検索に多大な時間と労力が必要不可欠である．これは，開発を阻む壁となり，プログラミングに対して嫌悪感を抱く要因となり得る．

このような問題に対して本研究では，プログラミング用語間の関係性から課題解決に関連する検索クエリを提示するシステムを提案する．

プログラミングに関して Web 検索を行うと，プログラミングに関する知識を記録・共有するためのサービス Qiita(注3)や IT エンジニア特化型 Q&A サイト teratail(注4)への投稿がヒットし，それらから有用な情報を得られることが多くある．そこで，プログラミング用語の情報源として，Qiita(注3) の投稿に紐付けられているタグに着目する．1 つの投稿において共起するタグは互いに関係すると仮定することで，プログラミング用語の関係性を抽出する．

提案システムでは，ユーザが目的と手段に関するクエリを一つずつ入力すると，その手段で目的を達成するためのより詳細なクエリが推薦される．例えば，Python で機械学習を行いたい場合には，「機械学習」を目的クエリとして，「Python」を手段クエリとして入力すると，「scikit-learn」や「Keras」などが推薦クエリとして提示される．ユーザはこれら推薦クエリを目的クエリとして用いることで再検索する．この過程を繰り返すことで，ユーザは課題解決に向けて学習すべき項目を絞り込んでいくことができる．

なお，推薦システムの基本方式として，大きく次の三つの方式がある；(a) 協調ベース推薦 [1]，(b) 内容ベース推薦 [2]，(c) 知識ベース推薦 [3] [4]．(a) および (b) の方式には，大量の履歴データが必要となる．また，(a) では新規ユーザおよび新規アイテムのコールドスタート問題がある．(b) では，新規アイテムのコールドスタート問題については解消されるものの，新規ユーザのコールドスタート問題は依然として残る．これらに対し，(c) は履歴データは必要とせず，新規ユーザに対しても推薦可能である．

知識ベース推薦には，制約ベース推薦方式と事例ベース推薦方式がある．制約ベース推薦は，ユーザが提示した制約を満たすアイテムを推薦する．事例ベース推薦は，ユーザが例示したアイテムや属性に類似するアイテムを推薦する．推薦されたアイテムを参考にしながら，ユーザが条件を修正（批評）していくことで，ユーザのアイテム探索を支援する．本研究では，履歴データをもたない新規ユーザにも推薦可能なシステムを想定しているため，知識ベース推薦方式に位置付けられる．

2. 関連研究

Boldi.P らの研究 [5] では，2 つの検索クエリに対するクエリ修正の種類として，Generalization，Specialization，Parallel Move，Mission Change，Error Correction の 5 つをあげている．

Generalization は，ユーザが現在の検索クエリからさらに検索範囲を拡げるクエリ修正である．例えば，ユーザが「Python PostgreSQL」から「Python データベース」に検索クエリを修正した場合である．このとき，検索条件は Python で特定のオブジェクト関係データベース管理システム PostgreSQL から，Python でデータベースに緩められた．

Specialization は，ユーザが現在の検索クエリからさらに検索

(注1)：https://www.google.co.jp/
(注2)：https://www.yahoo.co.jp/
(注3)：https://qiita.com/
(注4)：https://teratail.com/

範囲を絞り込むクエリ修正である．例えば，ユーザが「Python データベース」から「Python PostgreSQL」に検索クエリを修正した場合である．このとき，検索条件は Python でデータベースから，Python で特定のオブジェクト関係データベース管理システム PostgreSQL に絞り込まれた．

Parallel Move は，ユーザが検索のコンテキストは変えずに，他の視点から検索を行うクエリ修正である．例えば，ユーザが Web アプリケーションの開発のためにフレームワークを検索したことに続けて，サーバについて検索を行う等である．

Mission Change は，ユーザが検索範囲を大幅に変更するクエリ修正である．例えば，ユーザがプログラミングについて検索していたとき，訪問ページの広告に旅行情報が掲載されており，ユーザはその情報が気になったために旅行情報について検索する等である．

Error Correction は，検索クエリは変更されるが，ユーザの検索意図は変更されないクエリ修正である．例えば，スペルミスの修正や，異なる表現に書き換える場合である．

本研究では，ユーザがより深い検索を行う Specialization のクエリ修正に関して，クエリ推薦システムを提案する．

3. 定　　義

本稿において用いる記号を以下のとおり定義する：

$d_k \in D$:　記事.

$t_i \in T$:　タグ.

$D_i = \{d : d \ni t_i\}$:　タグ t_i を含む文書集合.

c_{ij}:　記事集合 D における t_i と t_j の共起頻度.

$T_i \subseteqq T$:　タグ t_i の関連タグ集合.

$q_1 \in T, q_2 \in T$:　入力クエリ.

R:　推薦クエリ集合.

4. シ ス テ ム

本章では，提案システムについて説明する．4.1 節でシステム概要について述べ，以下，細部について説明する．

4.1　システム概要

提案システムでは，二つのクエリ q_1, q_2 が入力されると，これらのクエリに共通するクエリ集合として，推薦クエリ集合 R が提示される．例えば，「Python を使って自然言語処理を行いたい」という要求を考える．この要求に対しては，q_1:「Python」，q_2:「自然言語処理」が入力クエリとなる．これらの入力クエリに対して，推薦クエリ集合として，例えば，「機械学習」や「scikit-learn」，「mecab」などが提示される．ユーザは，これらの推薦クエリを用いて Qiita(注3) の記事を検索するか，推薦クエリを次の入力クエリとして繰り返し推薦クエリを得ることができる．

図 1 は，システムのインタフェースイメージを示している．図では，入力クエリを q_1:「Python」，q_2:「自然言語処理」としたときの推薦例を示している．図のように，クエリ間の関連はグラフで表現される．グラフのノードはクエリを，リンクはクエリ間に関連があることを示している．入力クエリは両端に表示され，推薦クエリは中央に表示される．

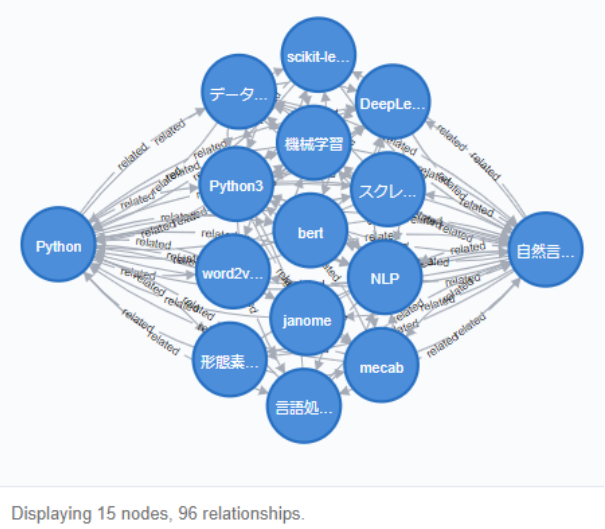

図 1　システムインタフェースのイメージ

4.2　Qiita の記事からのタグ集合の収集

Qiita は，「プログラミングに関する知識を記録・共有するためのサービス」(注5) である．Qiita の記事には，記事タイトル，関連タグ，本文等の情報が含まれる．Qiita API v2(注6) を用いることで，Qiita のデータを取得することができる．

本研究では，Qiita API v2 を用いて，Qiita の記事に登録されているタグ集合を収集する．Qiita の記事を $d_k \in D$ とする．収集したタグ集合を $T = \{t_1, t_2, \ldots, t_{|T|}\}$ とする．二つのタグ t_i, t_j が同一の記事に登録されている場合，タグ t_i と t_j は共起しているとし，記事集合 D におけるその共起頻度を c_{ij} とする．特に，タグ t_i について，$c_{ij} \geqq \alpha$ となるタグ t_j を t_i の関連タグとし，そのタグ集合を $T_i \subseteqq T$ と表す．

4.3　推薦クエリ集合の抽出

二つの入力クエリを q_1, q_2 とする．これらのクエリはタグ集合 T に含まれるタグの中から選択されるものとする．すなわち，$q_1 \in T, q_2 \in T$ となる．それぞれの入力クエリ q_i, q_j に関連するタグ集合をそれぞれ T_1, T_2 とする．このとき，T_1, T_2 の共通集合 $T_1 \wedge T_2$ を推薦クエリ集合 R とする．

4.4　ノイズタグの除去

Qiita に登録されているタグの中には，ごく限られた文書にしか現れないようなタグも含まれる．本稿では，このようなタグをノイズタグとよび，タグ集合から除去する．

タグ t_i を含む文書集合を $D_i = \{d : d \ni t_i\}$ とする．このとき，$|D_i| \leqq \beta$ となる t_i を低頻度タグとみなし，タグ集合 T から除去する．低頻度タグの例を以下に示す．

- 小規模開発
- 起業体験
- プログラムコンテスト
- 踊らせてみた
- ジャンク PC

(注5)：https://qiita.com/about
(注6)：https://qiita.com/api/v2/

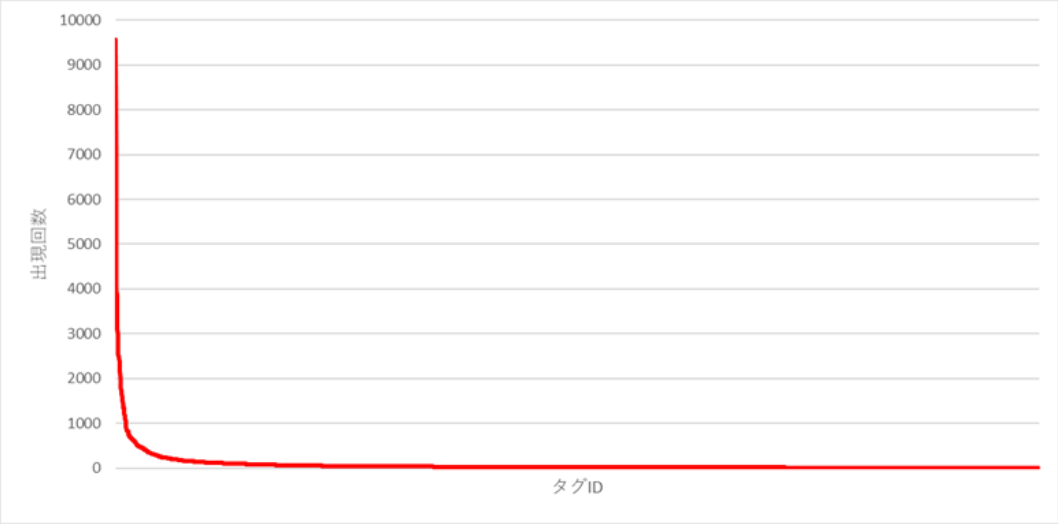

図 2　タグの出現頻度の分布．横軸はユニークなタグの ID を示し，縦軸は各タグの出現頻度を表す．横軸はタグの出現頻度の降順にソートしている．

5. 検　　証

5.1 データセット

本研究では，プログラミング用語の情報源として，Qiita(注3) の投稿記事に登録されているタグに着目する．投稿データの取得には，Qiita API v2(注6) を利用する．取得した投稿記事は次の通りである；

最も新しい記事の投稿日時： 2019/01/01 08:52:53
最も古い記事の投稿日時： 2017/12/12 01:04:08
記事の投稿件数： 108,057 件
全記事に登録されている延べタグ数： 300,128 件
ユニークなタグ数： 23,359 件

5.2 タグの出現頻度と内容の分析

4.4 節で述べたとおり，本研究では，タグの出現頻度に基づき，ノイズタグを検出する．ここでは，タグの出現頻度と内容の分析に基づき，低頻度タグ判定のための閾値 β を決定する．

図 2 は，記事に登録されている各タグの出現頻度の分布を表したものである．図の横軸はユニークなタグの ID を示し，縦軸は各タグの出現頻度を表す．横軸はタグの出現頻度の降順にソートしている．図 2 から，出現頻度の低いタグが大部分を占め，ロングテールの形状をなしていることがわかる．

出現頻度ごとのタグの内容について分析する．表 1 は出現頻度上位 10 件のタグを示している．表 1 のように，「python」や「javascript」，「ruby」など，プログラミングに関するキーワードが上位を占めた．

一方で，今回のデータセットにおいては，出現頻度が 1 のタグは 11,779 件あった．このタグの中には，「中学生」や「小学校」，「未経験入社」など，プログラミングに直接関連のないタグも多く含まれている．また，「めんどくさいことはプログラムにやらせよう」や「初期の c 言語：ほぼ忘れかけています。」というような，長めの文字列で汎用性の低いタグも含まれていた．このようなタグは本研究の目的においては，ノイズタグとして扱う．

表 1　タグの出現頻度　上位 10 件

タグ名	出現回数
Python	9,570
JavaScript	6,855
Ruby	4,150
Rails	3,974
PHP	3,852
AWS	3,822
Docker	3,097
Python3	2,977
Java	2,783
Android	2,540

以上の結果を踏まえ，本実験では，低頻度タグ判定のための閾値を $\beta = 1$ とする．

5.3 推薦クエリ集合の検証

4.3 節で述べた推薦クエリ集合の抽出手法の妥当性を検証する．本実験では，(a)「Python-自然言語処理」，(b)「Python-画像処理」，(c)「Python-Web」の各対を入力クエリとして与えたときの結果について考察する．それぞれのクエリ対を入力としたときの推薦クエリ集合を表 2 に示す．

a） Python-自然言語処理

表 2 のとおり，「Python-自然言語処理」を入力クエリとして与えたときの推薦クエリとして，「bert」や「word2vec」，「形態素解析」などが出力された．この結果から，Python により自然言語処理を行いたいという場合に，これらのタグが提示されることで，より具体的な記事を検索するための手がかりとなることが期待される．

b） Python-画像処理

表 2 のとおり，「Python-画像処理」を入力クエリとして与えたときの推薦クエリとして，「OpenCV」や「DeepLearning」などが出力された．「Python-自然言語処理」の結果と同様，Python

表 2　入力クエリ対に対する推薦クエリ集合

入力クエリ対	推薦クエリ集合
(a)Python-自然言語処理	データ分析, DeepLearning, 機械学習, Python3, スクレイピング, bert, scikit-learn, word2vec, NLP, janome, 形態素解析, mecab, 言語処理
(b)Python-画像処理	Python3, 機械学習, OenCV, DeepLearning, MachineLearning, C++, Unity, C#
(c)Python-Web	HTML, Java, google, Ruby, Django, PHP, JavaScript,Node.js, CSS, Mac, セキュリティ

により画像処理を行いたいという場合に，これらのタグが提示されることで，より具体的な記事を検索するための手がかりとなることが期待される．

一方で,「Python3」や「C++」,「C#」などのように，再検索のための手がかりとしては有用でないタグも含まれていた．

c） Python-Web

表 2 のとおり,「Python-Web」を入力クエリとして与えたときの推薦クエリとして,「Django」などが出力された．このように，Python による Web 開発に関して調べたいときには,「Django」で再検索することで，より具体的な記事を検索することができるということがわかる．また,「HTML」や「JavaScript」なども出力された．これは，先の目的においては，Python と併せて「HTML」や「JavaScript」についても調べる必要があるということを示唆するような結果といえる．

5.4 議　　論

提案システムにおいて，二つの入力クエリを仲介するタグの提示は可能であるものの，いくつかの問題点がある．本節では，その問題点について議論する．

5.4.1 同義語分類

5.1 節で取得したタグには同義語が登録されているものがある．例えば，以下のタグが挙げられる．

- Python，Python2，Python2.7，Python3
- GoogleCloudPlatform，gcp
- 機械学習，MachineLearning
- 人工知能，AI
- 自然言語処理，自然言語解析

例えば,「自然言語処理」と「自然言語解析」の場合,「自然言語処理」は出現頻度が 329 であったのに対し,「自然言語解析」のそれは 1 であった．今回の実験条件では,「自然言語解析」はノイズタグとして扱われてしまう．

これらのようなタグを同義語として判別し，いずれのタグを入力クエリとしても，同一の推薦結果を返す必要がある．また，推薦クエリからも入力クエリとの同義語や推薦クエリ内の同義語を除去する必要がある．

5.4.2 ノイズタグの検出

本研究では，4.4 節で低頻度タグをノイズタグとして検出する方式を用いたが，低頻度タグにおいてもプログラミングに関連するタグは多く含まれる．今後は，さまざまなノイズタグの検出手法を導入し，適切な検出方法を選択する必要がある．

5.4.3 推薦クエリの絞込み

提案システムにおいて，目的クエリと手段クエリ共に多数の関連タグをもつ用語とした場合，推薦クエリが膨大となり，ユーザによる検索クエリの選定が困難になる．例えば,「Python-機械学習」を入力クエリ対とすると，推薦クエリ数は約 70 にも上る．そのため，システム側で推薦アイテムの絞込みを行う必要がある．

5.4.4 再検索における推薦クエリの再出現

提案システムでは，推薦クエリを目的クエリとして用いることで再検索を繰り返し，ユーザが課題解決に向けて学習すべき項目を絞り込むことを目指している．しかし，現状では再検索を行うと前回の推薦クエリも再出現される問題がある．例えば,「Python-機械学習」を入力クエリ対とすると，推薦クエリとして「scikit-learn」が提示されるが，推薦クエリを目的クエリとして「Python - scikit-learn」で再検索すると，推薦クエリとして「機械学習」が再度提示される．ユーザにとっての利便性を考慮したとき，一度推薦クエリとして出力されたタグは，同一セッションにおいては出力されないようにすることが重要である．

5.4.5 検索クエリとタグの関連付け

提案システムにおいて，現状での入力はタグとの完全一致でのみ受け付ける．しかし，ユーザがタグと完全一致するクエリを入力しない可能性も大いにあるため，曖昧な検索クエリをタグと関連付けする必要がある．例えば，検索クエリを「クラウド」とした場合,「AWS」や「GoogleCloudPlatform」等のタグ名を関連付ける必要がある．

6. おわりに

本研究では，プログラミング初学者に対して課題解決に関連するクエリを推薦するシステムを提案した．データセットとしては，Qiita(注3) の投稿データを用いた．提案システムでは，データセットのタグからプログラミング用語間の関係を抽出を行った．ノイズタグの除去においては，tf-idf およびタグ間距離行列を DBSCAN でクラスタリングする 2 パターンを検証した．プログラミング用語間の関係性格納および検索には，グラフデータベース Neo4j を用いた．ユーザは目的クエリと手段クエリを入力とすることで，それらを仲介するクエリが推薦される．現状において，仲介するクエリの推薦は可能であるものの，5.4 で述べたように，推薦や検索に関する複数の問題が存在するため，今後，これらを解消する必要がある．

謝　　辞

本研究は JSPS 科研費 19K12567 の助成を受けたものです．ここに記して謝意を表します．

文　　献

[1] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. Evaluating collaborative filtering recommender systems. *[ACM]ACM Transactions on Infor-*

mation Systems, Vol. 22, No. 1, p. 553, 2004.

[2] Ivn Cantador, Alejandro Bellogín, and David Vallet. Content-based recommendation in social tagging systems. In *[RecSys2010]Proceedings of the 4th ACM conference on Recommender systems*, p. 237, 2010.

[3] F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor. *Recommender Systems Handbook*. Springer, 2011.

[4] Marius Kaminskas, Ignacio Fernández-Tobías, Francesco Ricci, and Ivn Cantador. Knowledge-based music retrieval for places of interest. In *Proceedings of the second international ACM workshop on Music information retrieval with user-centered and multimodal strategies - MIRUM '12*, pp. 19–24, New York, New York, USA, 2012. ACM Press.

[5] Carlos Castillo Sebastiano Vigna Paolo Boldi, Francesco Bonchi. " From Dango to Japanese Cakes ": Query Reformulation Models and Patterns. In *In Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, p. Volume 01, 2009.