F-004

ユーザの検索情報を用いた商品レコメンドの精度改善

田口 拓明[†1,a)]　日暮 立[†1,b)]　清水 伸幸[†1,c)]　田島 玲[†1,d)]

概要：推薦システムはサービスを利用したことがない新規ユーザへの商品のレコメンドが困難という問題（コールドスタート問題）を抱えている．本研究では，ショッピングサービスの新規ユーザへの商品のレコメンドのために，多くのユーザに紐づく検索情報を利用した．その結果，商品のレコメンドの精度を向上させることができ，ユーザの商品に対するクリックをより獲得することができた．

キーワード：機械学習，レコメンド，検索データ，自然言語処理，ニューラルネットワーク

Recommendation System Performance Improvement with Search Information

Hiroaki Taguchi[†1,a)]　Tatsuru Higurashi[†1,b)]　Nobuyuki Shimizu[†1,c)]　Akira Tajima[†1,d)]

Abstract: The recommendation system has a problem of difficulty in recommending products to new users who have never used the service (cold start problem). In this study, we used to search information associated with many users to recommend products to new users of the shopping service. As a result, we were able to improve the accuracy of product recommendations and obtain more clicks on products from users.

Keywords: Machine learning, recommendation, search data, natural language processing, neural networks

1. 背景

多くの企業では，ユーザの嗜好を予測する推薦システムを用いて売上の向上に繋げている．電子書籍販売サイトの ebookjapan では，推薦システムを利用して電子書籍をレコメンドしており，日々レコメンドの性能を向上させてユーザが直近で購買したくなりそうな商品をうまく推薦することで，売上を向上させている．

しかし，その推薦システムは新規ユーザが ebookjapan に訪れた際に商品のレコメンドが困難という問題（コールドスタート問題）を抱えている．例えば，本研究で利用する ebookjapan のデータのみを利用して電子書籍の推薦を行うと，新規ユーザとデータ数が少ないユーザに対してはユーザを理解する情報が少なく，適切なレコメンドを行いにくい．

そこで，本研究ではそのコールドスタート問題を解決するために，ebookjapan のデータとそれとは異なるドメインのデータである Yahoo!検索の検索データを利用してレコメンドを行った．検索データを利用した理由は，Yahoo!サービスを利用している多くのユーザが Yahoo!検索を利用しており，そのユーザ数は ebookjapan のユーザ数よりも多いため，ebookjapan のデータが少ないユーザにも検索データが紐付いているケースが多いと考えられるためである．

本研究では，ebookjapan の推薦システムに検索データを加えてレコメンドの性能向上を図った．その結果，ebookjapan 上のデータが少ないユーザ対して，レコメンド性能の向上が顕著に見られ，レコメンド結果を本番配信した際にはユーザの商品に対するクリックと購買をより獲得できため，本手法の有効性が示された．

†1 現在，ヤフー株式会社
Presently with Yahoo Japan Corporation
a) htaguchi@yahoo-corp.jp
b) thiguras@yahoo-corp.jp
c) nobushim@yahoo-corp.jp
d) atajima@yahoo-corp.jp

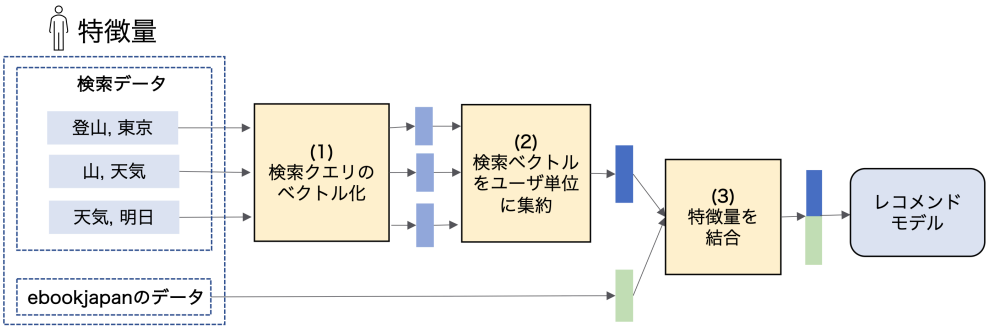

図 1　提案手法の概要

2. 提案手法

この章では，検索データを用いたユーザの特徴量の作成手法と，レコメンドタスクを解決する機械学習モデル (以下，レコメンドモデルと呼ぶ) に，そのユーザの特徴量を適応する方法について述べる．手法の概要は図 1 に示すように下記の 3 つのステップに分かれる．

(1) ユーザの検索クエリをユークリッド空間に埋め込むクエリエンコーダを作成し，それを利用して検索クエリをベクトル化する

(2) 検索クエリのベクトルをユーザ毎に集約して，ユーザ単位の検索ベクトルを作成する

(3) ebookjapan のデータから取得できる特徴量とユーザ単位の検索ベクトルの 2 つを結合して，レコメンドモデルの特徴量として利用する．

2.1 検索クエリのベクトル化

ユーザの検索特徴量は，検索クエリをユークリッド空間に埋め込むクエリエンコーダを作成し，それを使って検索クエリをベクトル化して利用した．クエリエンコーダーは，リカレントニューラルネットワークをベースとし，DSSM[1] のフレームワークを利用して次のクエリを予測する学習を行うことで作成した．次のクエリを予測する学習は，クエリ Q は正の例（次のクエリ）とその他のクエリ Q'_k を負の例 (ランダムにサンプリングした疑似の負の例) をリカレントニューラルネットワークベースで構成されたクエリエンコーダ f を用いてユークリッド空間に埋め込み，損失関数 ℓ を式 (1) に設定して行った．その際の類似度スコア R は式 (2) のように定義される．

$$\ell\left(Q, Q'_0\right) = -\log \frac{\exp R\left(Q, Q'_0\right)}{\sum_k \exp R\left(Q, Q'_k\right)} \tag{1}$$

$$R\left(Q, Q'_0\right) = \cos\left(f(Q), f\left(Q'_0\right)\right) \tag{2}$$

式 (1) の DSSM による損失関数に加えて言語モデルの損失関数も利用し，それらを用いて交互に最適化することで全体の学習を行った．上記の方法で学習されたクエリエンコーダを利用して，検索クエリから 128 次元の検索ベクトルを作成した．

2.2 ユーザ単位の検索ベクトルの作成

ユーザの検索特徴量は検索クエリが時間順で並んでいるため，テキストシーケンスの局所的な情報を保持して埋め込みベクトルを合成できる SWEMs[2] の SWEM-hier を利用した．SWEM-hier は，まず連続する単語からなるローカルウィンドウにて，各ローカルウィンドウ内の単語の埋め込みベクトルに対して average-pooling を行い，その結果を global max-pooling することで，階層的な pooling を行って埋め込みベクトルを合成する手法である．その SWEM-hier を利用して，ユーザの時間順に並んだ検索ベクトルを合成してユーザ単位の検索ベクトルを作成した．そのユーザ単位の検索ベクトルは，2.1 節の検索ベクトルの次元と同じ 128 次元である，そこから，正の値の要素を抽出した 128 次元のベクトル (負の値は 0 に置換) と負の値の要素を抽出した 128 次元のベクトル (正の値は 0 に置換) の 2 つを作成し，それらを結合して最終的に 256 次元のベクトルを作成した．

3. 評価実験

ユーザ単位の検索ベクトルの有効性を示すために，レコメンドモデルにユーザ単位の検索ベクトルを適応した場合と適応しない場合のレコメンド精度を比較する実験を行った．レコメンドの精度を測るための評価指標は，MRR と recall@k を利用した．

レコメンドタスク　レコメンドはサービスの商品ページに訪れたユーザが，次にどのような商品を閲覧しそうかを予測して，その結果をユーザに提示することで行う． そのレコメンドを行う機械学習モデルは，特徴量から商品 ID を予測する学習を行って作成した．

特徴量 ebookjapan のデータに関する特徴量は，ユーザの閲覧した商品 ID とそれらの商品のメタデータ (著者，タイトル，ジャンル)，ユーザの性別の情報を利用しており，検索データに関する特徴量には，ユーザの検索クエリから 2.2 節で述べたユーザ単位の検索ベクトルを作成して利用した．

レコメンドモデル レコメンドのための機械学習モデルは，中間層が 1 層のニューラルネットワークで構築されており，特徴量から商品 ID を予測するマルチクラス分類問題が解けるモデルとなるように設計されている．

データセット データは Yahoo!検索の検索データと ebookjapan のデータを利用した．検索データの集計期間は 2020 年 8 月 3 日から 2020 年 9 月 1 日までとし，ebookjapan のデータの集計期間は 2020 年 9 月 2 日から 2020 年 9 月 9 までとした．検索データの集計期間が ebookjapan のデータの集計期間と被らないようにしている理由は，モデルの学習時にリークが発生しないようにするためである．

次に，モデルの学習時と推論時のデータセットの使い方の詳細を図 2 に示す．ebookjapan のデータは時間軸を考慮して前後で分割して学習データとテストデータを作成した．モデル学習時は ebookjapan の学習データとユーザ単位の検索ベクトルを利用して学習を行い，推論時は ebookjapan のテストデータと学習時に利用したユーザ単位の検索ベクトルを利用した．設定した ebookjapan のテストデータのユーザの商品閲覧回数毎のデータ数は図 3 となっている．ユーザの商品閲覧回数の最大値が 10 回となっているのは，システムが閲覧履歴を 10 個までしか保持しない仕様となっているためである．

また，レコメンドモデルにユーザ単位の検索ベクトルを入れる場合に，ebookjapan のデータは紐づくが検索データは紐付かないユーザが存在する．そのユーザには学習データに存在するユーザ単位の検索ベクトルを各次元で平均して利用した．

3.1 実験結果

初めに，ユーザ単位の検索ベクトルを適応した場合と適応しない場合でレコメンドタスクの評価指標の数値を比較した結果を表 1 に示す．表 1 よりユーザ単位の検索ベクトルありの方が，すべての評価指標 (MRR，recall@1，recall@5) の数値が高いため，ユーザ単位の検索ベクトルをレコメンドモデルに適応することでレコメンド精度を改善できると言える．

次に，ebookjapan 上でのデータが少ないユーザが ebookjapan に訪れた場合に，レコメンドが適切に出来ているかどうかを確認するために，ユーザの商品閲覧回数毎に recall@1 を調査して，ユーザの商品閲覧回数が少ない場合 (0 回，1 回，2 回) とそうでない場合のレコメンドの精度を計測した．その結果を図 4 に示す．図 4 より，閲覧回数が 0 回のときは検索ベクトルなしの recall@1 が 0.005 に対して検索ベクトルありは 0.024，閲覧回数が 1 回のときは検索ベクトルなしの recall@1 が 0.061 に対して検索ベクトルありは 0.062，閲覧回数が 2 回のときは検索ベクトルなしの recall@1 が 0.064 に対して検索ベクトルありは 0.067 となり，ユーザの商品閲覧回数が少ない場合 (0 回，1 回，2 回) にユーザ単位の検索ベクトルを適応すると，レコメンド精度を改善させることができた．特に閲覧回数が 0 回の場合は，式 (3) に示すレコメンド精度の改善率が 4.8 倍になっている．そのため，閲覧履歴の全く存在しない新規ユーザの場合にユーザ単位の検索ベクトルを適応してレコメンドを行うのは効果的であると言える．また，図 3 より，ユーザの商品閲覧回数が少ない場合 (0 回，1 回，2 回) のデータの割合は全体の 16.3%ほどあり，そのデータに対してレコメンド精度の改善が出来ている．

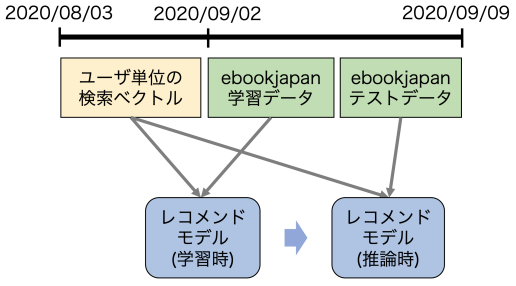

図 2 モデルの学習と推論時のデータセットの利用方法

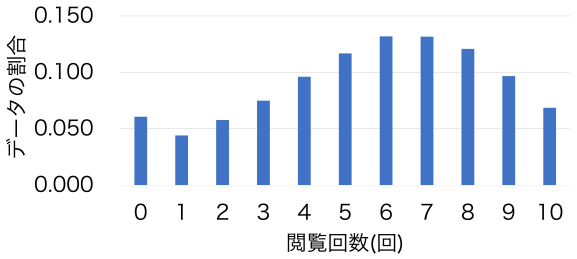

図 3 テストデータのユーザの商品閲覧回数毎のデータ数

表 1 レコメンドモデルにユーザ単位の検索ベクトルを適応した場合とそうでない場合のレコメンド精度の比較

	検索ベクトルなし	検索ベクトルあり
MRR	0.085	**0.087**
recall@1	0.035	**0.036**
recall@5	0.119	**0.120**

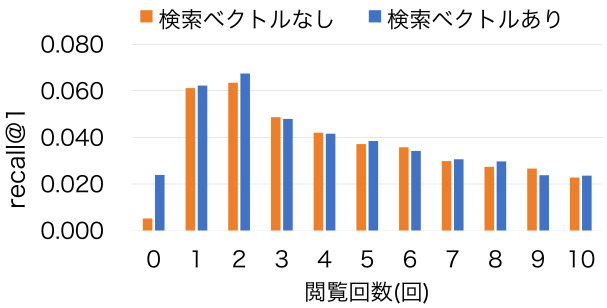

図 4 ユーザの商品閲覧回数毎にレコメンド精度を比較した結果

$$\text{レコメンド精度の改善率} = \frac{\text{検索ベクトルありの recall@1}}{\text{検索ベクトルなしの recall@1}} \quad (3)$$

4. 本番配信実験

評価実験で本手法の有効性が確認できたため，本番環境

の A/B テストにてユーザ単位の検索ベクトルを実サービスに適応した際のレコメンドの効果を調査した．その効果検証では，本番環境で利用しているレコメンドモデルに，ユーザ単位の検索ベクトルを適応した場合と適応しない場合のレコメンドの効果を計測して比較した．

効果の比較にはランダム化比較試験を利用した．ランダム化比較試験では ebookjapan に訪問した一部のユーザに対して，50%の確率でユーザ単位の検索ベクトルをレコメンドモデルに適応した場合のレコメンド結果を出し，50%の確率で適応しなかった場合のレコメンド結果を出すことで，各レコメンド結果をランダムで出し分けて，その際のレコメンドの効果を計測した．レコメンドモデルは，3 章の評価実験と同様のモデルを利用し，A/B テストの期間 (2021/8/13〜2021/8/26 の 14 日間)，毎日学習を行ってモデルを定常的に作成した．モデルの学習時に利用する ebookjapan のデータの集計期間は，モデルが学習するタイミングより前の 1 週間分のデータを利用し，学習時に利用する検索データは 1 週間前よりも以前に作成された最新のユーザ単位の検索ベクトルを利用して，モデルの学習を行った．これにより，3 章の評価実験と同様に，ebookjapan のデータの集計期間と検索データの集計期間を被らないようにした．

レコメンドの効果には，レコメンド結果を掲載した際のユーザのクリック率とコンバージョン率を用いた．そのクリック率とコンバージョン率から，ユーザ単位の検索ベクトルを適応した場合のレコメンド効果の改善率を計測した．まず，クリック率を式 (4)，コンバージョン率を式 (5) にそれぞれ示す．

$$\text{クリック率 (CTR)} = \frac{\text{ページ全体のクリック数}}{\text{ページの閲覧数}} \tag{4}$$

$$\text{コンバージョン率 (CVR)} = \frac{\text{ページ閲覧後の購入件数}}{\text{ページの閲覧数}} \tag{5}$$

次に，クリック率の改善率 (%) を式 (6)，コンバージョン率の改善率 (%) を式 (7) にそれぞれ示す．

$$\frac{\text{検索ベクトルありの CTR - 検索ベクトルなしの CTR}}{\text{検索ベクトルなしの CTR}} \times 100 \tag{6}$$

$$\frac{\text{検索ベクトルありの CVR - 検索ベクトルなしの CVR}}{\text{検索ベクトルなしの CVR}} \times 100 \tag{7}$$

クリック率は，作品詳細ページと書誌詳細ページの 2 つのページにレコメンド結果を掲載した際の数値を測定した．作品詳細ページとは，電子書籍のシリーズをまとめたページであり，書誌詳細ページとは電子書籍 1 冊に対するページである．また，式 (4) のクリック数はレコメンド結果をクリックした回数以外も含む．

コンバージョン率は，レコメンド結果全体に対する数値とユーザのセグメント毎 (新規ユーザと既存ユーザ) に分けた場合の数値を測定した．新規ユーザとは，ebookjapan のサービスで商品を購入したことのないユーザであり，既存ユーザとは商品を購入したことのあるユーザである．また，式 (5) の購入件数はユーザのページの閲覧行動と購入行動が同日に行われた場合の件数を計測した．

4.1 実験結果

初めに，クリック率の改善率を確認した．書誌詳細ページのクリック率の改善率は +1.71% であり，検索ベクトルなしのクリック率と検索ベクトルありのクリック率から有意差を確認したところ，統計的に有意 ($p < 0.05$) であることが確認できた．そのため，ユーザ単位の検索ベクトルをレコメンドモデルに適応することで，ユーザがクリックしやすくなるレコメンドができたと言える．また，作品詳細ページのクリック率の改善率は −0.93% ではあったが，検索ベクトルなしのクリック率と検索ベクトルありのクリック率から有意差を確認したところ，有意差は確認できなかった．

次に，コンバージョン率の改善率を確認した．コンバージョン率の改善率は，ユーザ全体では +1.36%，新規ユーザでは +4.17%，既存ユーザでは +0.36% となっていたため，ユーザ単位の検索ベクトルをレコメンドモデルに適応するとコンバージョン率が改善した．また，ユーザ全体の場合にて検索ベクトルなしのコンバージョン率と検索ベクトルありのコンバージョン率から有意差を確認したところ，統計的に有意 ($p < 0.05$) であることが確認できた．この結果から，ユーザ単位の検索ベクトルを利用することで，ユーザが商品を購買しやすくなるレコメンドができたと言える．

5. 結論

本研究では，ebookjapan の推薦システムに検索データを加えてレコメンドの性能向上を図った．その結果，ebookjapan のデータが少ないユーザ対して，レコメンド性能の向上が顕著に見られて，レコメンド結果を本番配信した際にはユーザの商品に対するクリックや購買をより獲得できため，本手法の有効性が示された．

謝辞
本番配信実験でご協力いただいた ebookjapan に勤められている皆様とレコメンド技術の開発部署の皆様に感謝申し上げます．

参考文献

[1] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. Learning deep structured semantic models for web search using clickthrough data. CIKM, 2013.
[2] Dinghan Shen, Guoyin Wang, Wenlin Wang, Martin Renqiang Min, Qinliang Su, Yizhe Zhang, Chunyuan Li, Ricardo Henao, and Lawrence Carin. Baseline needs more love: On simple word-embedding-based models and associated pooling mechanisms. ACL, 2018.