Vol.2020-GN-110 No.11
2020/3/17

口コミ解析と好み診断に基づいた旅行先推薦

市村 哲[1]　上石 萌恵[1]　堀口 莉里花[1]

概要：旅行先を考える際，旅行サイトを利用することが多い．しかしながら，掲載されている情報量が多すぎて旅行先を決めるのが難しいという問題がある．また，通常，各旅行先ごとに口コミが登録されており，行きたい旅行先がまだ決まっていない人には使いにくい面がある．そこで著者らは，旅行計画を立てる際に多くの旅行情報を閲覧しなくても手軽に旅行先を選ぶことができるようにすることを研究の目標に定め，この目標に沿い口コミ解析と好み診断に基づいた旅行先推薦 Web サービス「旅ゲーター（tabi-gator）」を開発した．旅ゲーターは，機械学習しておいた旅行サイトの口コミデータセットから，ユーザの好みを診断する幾つかの質問を自動的に作成する．各質問にユーザがどのように回答したかに基づいてユーザの好みに適した旅行先を推薦する．

Travel Plan Recommendation Based on Review Analysis and Preference Diagnosis

Satoshi Ichimura[1]　Ageishi Moe[1]　Horiguchi Ririka[1]

Abstract:
When making our travel plans, we often use travel sites. However, there is a problem that it is difficult to decide a travel plan because the amount of information is too large. Also, each user's review is usually registered for each destination, which is difficult to use for those who have not yet decided the destination they want to go to. Therefore, the authors set a research goal to make it possible to select a travel destination without having to browse a lot of travel information when making a travel plan, and based on this goal, based on user's review analysis and preference diagnosis. We have developed a web service for recommending travel plans, called Tabi-gator (Travel navigator in English). Tabi-gator automatically creates several questions to diagnose user preferences from the users' review dataset with machine-learning technology. A travel plan suitable for the user's preference is recommended based on how the user answered each question.

1. はじめに

インターネットを利用して旅行計画を立てることが一般的になっている．旅行計画を立てる際にまず何を利用するかとの質問に対し，約 85 %の人がインターネットを利用すると回答したとの報告がある [1].

旅行先を検索することができるサイト（通称，旅行サイト）では，ユーザはまず地域や旅行目的などの条件を入力して検索するが，それと同時に検索された旅行先について他の人が書き込んだ感想（通称，口コミ）を参考にすることが多い．実際に訪れたであろう人たちの感想は非常に有用であり，旅行代理店などから提供される写真や文面以外の多くの情報を知ることができる．またその旅行先を訪れた後，旅先の情報を口コミとして書き込むことも一般的になっている．また旅に関する質問を書き込むと他の人から有益な回答を得られることが多い．その回答も旅行サイトの口コミ情報として活用されることとなり，そうした口コミを利用して実際に行く場所を決めるユーザは多い．例えば日本最大級の旅行サイト 4travel では，ユーザ同士が旅行先について口コミを投稿して共有したり，旅の思い出を旅行記にまとめて掲載したりすることができ，「みんなでつくる旅行ガイド」として旅行先別にそれらの書き込みを登録することができる．

しかしながら，旅行サイトに登録されている旅行先は非

[1] 大妻女子大学社会情報学部情報デザイン専攻
12 Banchi, Sanbancho, Chiyoda-ku, Tokyo, 102-8357 Japan

ⓒ 2020 Information Processing Society of Japan

常に多く，かつ，各旅行先について書き込まれた口コミ情報も大量であることが普通である．旅行の計画を難しいと感じたことがあるかとの質問に対し約 7 割の人がそう思うまたはややそう思うと回答しており，その理由の第 1 位が「情報や選択肢が多いため (66.7 %)」，第 2 位が「すぐに好みの情報がみつからないため (52 %)」であったと報告されている [1]．数件程度の口コミであれば一つ一つ見ていくこともできるが，数百件の口コミを読むとなると膨大な時間がかかってしまい，すべてを読むのは現実的ではない．そのため適当に目についた口コミをいくつか読むだけで旅行先を選定しなければならないのが実情である．

さらに，通常の旅行サイトにおいては，旅行先ごとに口コミが登録されており，行きたい旅行先がまだ決まっていない人には使いにくい面がある．実際上述した 4travel においても，口コミを見る場合は旅行先候補地を何度も変えながら検索を繰り返す必要がある．このようなことから，旅行計画を立てることに対してストレスを感じ，旅行に行く前に疲れてしまう人も少なくない．

以上の問題に着目し，旅行計画を立てる際に多くの旅行情報を閲覧しなくても手軽に旅行先を選ぶことができるようにすることを研究の目標に定め，この目標に沿い，口コミ解析と好み診断に基づいた旅行先推薦 Web サービス「旅ゲーター（tabi-gator）」を開発した．旅ゲーターでは，ユーザはおおまかな地域を指定した後は，システムから出される幾つかの質問に答えるだけで，自分にあった旅行先が推薦されるようになっている．

旅ゲーターは，予め機械学習しておいた旅行サイトの口コミデータセットから，ユーザの好みを診断する幾つかの質問を自動的に作成する．ユーザは，質問に回答する際システムが提示した選択肢から好みのものを選ぶだけでよい．各質問にユーザがどのように回答したかに基づいてユーザの好みに適した旅行先が推薦される．

2. 関連研究

旅行サイトには大量の旅情報が掲載されているが，それが故に必要な情報にたどり着くのに時間がかかる問題がある．同様な問題がインターネット上の他のサービスについても生じており，これらの問題に対する関連研究について以下に述べる．

大山ら [2] は，ゲームレビューサイトに登録されている大量のレビュー文章を Word2Vec[15] を用いて機械学習し，ユーザがキーワードを入力した際に，類似度が高いゲーム名を出力するシステムを提案している．Word2Vec は大量のテキストデータを解析して単語をベクトル化する手法である．単語をベクトル化することで，ベクトル間の距離を比較して，その単語に近い単語，すなわち類似した単語を出力することや，単語と単語の類似度を求めることができる．一般的に Word2Vec は中間層と出力層の 2 層からなるニューラルネットワークで構成されている．

梅本ら [3] は，料理レシピにおける代替食材の推薦のために Word2Vec と Doc2Vec[13][14] を用いる手法を提案している．料理の食材を Word2Vec を用いて機械学習すると共に，レシピサイトに登録されている大量のレシピ文章を Doc2Vec を用いて機械学習している．Word2Vec が入力として単語列を受け取るのに対し，Doc2Vec は文章 ID を併せて入力することで，単語のベクトル化だけでなく文章のベクトル化が可能である．これにより，類似した文書を出力することや，文章と文章の類似度を求めることができる．当該システムでは，作りたい料理のレシピ（対象レシピ）と，不足している食材（対象食材）を入力すると，類似料理レシピに使用されている食材と対象食材の類似度を Word2Vec によって計算し，また，対象レシピと類似するレシピを Doc2Vec によって計算し，両方の計算結果にもとづいて類似度を求め代替食材の候補を出力している．Word2Vec 同様に Doc2Vec も中間層と出力層の 2 層からなるニューラルネットワークで構成されている．

谷本ら [4]，平山ら [5] および中野ら [6] は，通販サイトにおける大量の投稿レビューを自然言語処理によって解析し，商品に対するユーザの意見を機能ごとに自動的に整理する方法を提案している．文の係り受け構造に着目して係り元（ユーザの意見）および係り先（商品の機能）のペアを抽出し，どの意見がどの機能に対する意見なのかをグラフ表示する機能や，ユーザの意見がポジティブなものネガティブなものかを日本語評価極性辞書を用いて解析しその統計情報をグラフ表示する機能を実装している．また紀本ら [7] は映画レビューサイトを対象として本手法を適用している．

一方で長い文章を短い文章に要約することによって，読むのにかかる時間を短くしようとする研究が存在する．

著者らは過去において，美味しい料理が食べられる料理店を手早く探すことができる Web サービス「食探」を開発した．ユーザはどのような料理が食べたいかに関する自分の希望や質問を日本語自然文で入力する．するとシステムは，予め機械学習しておいたグルメサイトの口コミデータセットから，ユーザの入力文に類似した口コミ文章を検出すると共に，そのような口コミが多い飲食店を抽出する．加えて，料理に関係する口コミ文章を優先的に抽出し，口コミ文章から抽出した料理に関する評価文を短い要約文で表示する機能を備えており，投稿レビュー文章に特化した文章要約手法となっている．しかしながら，通常のグルメサイトにおいては各店ごとに口コミが登録されているため，店が決まる前の段階では，店の候補を色々と変えながら検索を繰り返す必要があった．

また ERKAN[8] らは，文章中の文や単語の関係性を元に文章を要約する LexRank 法を提案している．文章の中で，多くの文に類似する文は重要文であり，かつ，重要文

に類似した文は重要文であるという考え方に基づいて重要箇所を抽出し提示する．投稿レビュー以外の一般的な文章に対しても汎用的に使用できる文章要約手法であるが，投稿レビュー文章に特化した要約手法に比べ希望どおりの要約が得られない可能性が高いという問題がある．

3. 提案

旅行計画を立てる際に多くの旅行情報を閲覧しなくても手軽に旅行先を選ぶことができるようにすることを研究の目標に定め，この目標に沿い，好みの旅行先を手早く探すことができる Web サービス「旅ゲーター」を開発した．旅ゲーターは，予め機械学習しておいた旅行サイトの口コミデータセットからユーザの好みを診断する幾つかの質問を自動作成する．このとき，各質問にユーザがどのように回答したかに基づいてユーザの好みに適した旅行先が推薦されるようになっている．質問に回答する際ユーザは，システムが提示した選択肢から好みのものを選ぶだけでよいため，希望する旅行先を手早く簡単に探すことができる．

旅ゲーターの Web ページを表示すると，選択肢 (ボタン) が表示される．ユーザは自分が好みの選択肢を選んでボタンを押す．すると，次の選択肢が現れるので，再度好みの選択肢を選んでボタンを押す．この操作を数回繰り返すと最後におすすめ旅行先とその旅行先についての口コミが表示される．

図を用いて説明する．旅ゲーターの Web ページを表示すると図 1 の画面が Web ブラウザに表示される（選択肢数 7 の場合の例）．これらの選択肢からユーザが「桜」を選んだとすると，次の質問画面の図 2 が表示される．ユーザが選択した履歴は「あなたの好み：」に表示される．ここでユーザが「子供」を選んだとすると図 3 が現れ，おすすめ旅行先が「東京ドイツ村」であることを示すと共に，最初の選択肢表示に戻るようになっている．なお，おすすめ旅行先が確定するまで図 2 のような質問画面が繰り返し表示される．

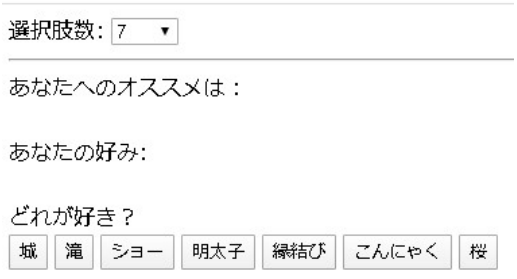

図 1　システム動作画面 (ステップ 1)

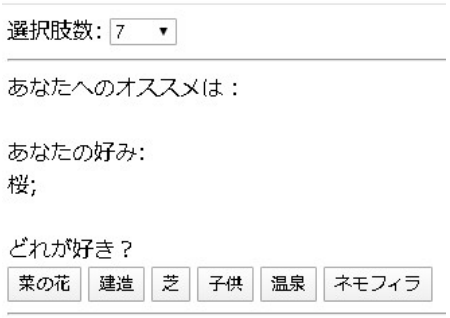

図 2　システム動作画面 (ステップ 2)

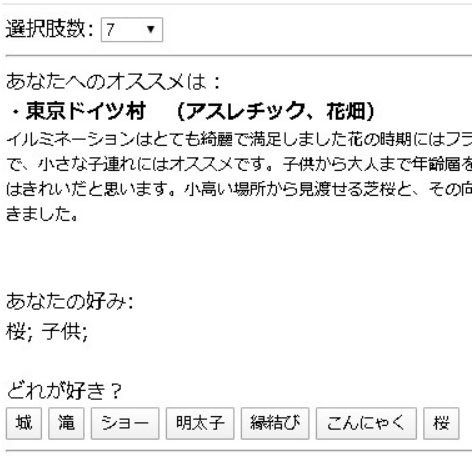

図 3　システム動作画面 (ステップ 3)

4. 実装

関東地方の有名観光スポット 70 地点を対象とした試作システムを実装した．各観光スポットの口コミデータ（各スポットの口コミ数は平均 7 件）は 4travel から取得して用いた．本システムは Python[17] で実装されたサーバプログラムと，JavaScript で実装されたクライアントプログラムから構成されている．

実装した主な機能は，(1) 口コミ文章のベクトル化，(2) 口コミ文章のクラスタリング，(3) 各クラスタの重要語抽出，(4) 選択クラスタの再帰的再分割，である．

4.1 機能説明

(1) 口コミ文章のベクトル化：全旅行先の全口コミ文章を Doc2Vec に入力して機械学習し，文書をベクトル化する．

具体的には，1 つの旅行先について書き込まれた多数の口コミを 1 つの文章にまとめて文章 ID を付与し，すべての旅行先についての文章（旅行先毎にまとめられた口コミ文章）と文書 ID（旅行先 ID に相当）を Doc2Vec に入力する（口コミ文章に加えて旅行代理店が作成するような旅行案内文を追加して入力しても構わない）．本実装では Python のライブラリである gensim[16] に含まれる Doc2Vec を用い，Doc2Vec のモデルは PV-DM(Paragraph Vector with Distributed Memory)，ベクトルは 100 次元，入力に与える単語列の長さである window は 5 として利用した．

Doc2Vec に入力する文章が日本語の場合は，分かち書き表記された文章を入力する必要があり，そのために MeCab の分かち書き機能を利用している．MeCab を用いる際，分かち書きされるべきでない固有名詞などを MeCab 標準辞書に多数追加した mecab-ipadic-NEologd [12] を利用した．

なお，文書をベクトル化する他の方法としては TF-IDF を用いた方法があるが，単語の出現回数のみを用いるために単語の共起関係が失われるデメリットがある．そこで今回は，単語の共起関係をベクトルに反映できる Doc2Vec を用いた手法を採用することとした．

(2) 口コミ文章のクラスタリング：(1) で作成された文章ベクトルを K 平均法 (K-Means) によって所定個のクラスタに分割する．K 平均法は非階層的クラスタリング手法の 1 つであり，類似度の高いベクトルをまとめ，指定した個数のクラスタに分類する方法である．今回の実装では gensim ライブラリに含まれる K-Means の実装を利用した．作成するクラスタ数（選択肢数）はユーザが自由に選べるようにした．

(3) 各クラスタの重要語抽出：(2) で作成されたクラスタごとに，TF-IDF 法を用いて重要語を抽出する．TF 値は文書内でのある単語の出現頻度であり，その単語の出現回数が多いほど値は大きくなり，出現回数が少ないほど値は小さくなる．IDF 値は文書集合の中のある単語が含まれる文書の割合の逆数であり，その単語が他の文章にも多く出現しているほど IDF 値は小さくなり，他の文章にあまり出現していないほど値は大きくなる．TF 値と IDF 値を掛け合わせた TF-IDF 値が大きい単語ほど，各クラスタ内で重要度が高い単語であるとみなせる．

TF-IDF 値を計算する際に gensim ライブラリに含まれる TF-IDF モジュールを利用している．(2) で作成された各クラスタから TF-IDF 値が最大となった単語（各クラスタを代表する重要単語）を抽出し，ユーザに選択肢として示す．具体的には Web ブラウザ上に選択肢ボタンとして表示される．

なおストップワード除去処理のために，Web 検索研究支援プログラミングライブラリ SlothLib[9] を用いたワード除去処理，および，連続した数字を 0 に置き換える処理などを施した．

また今回の実装では，旅行サイトに登録された口コミデータセットからネガティブでない文（ここで文とは句点から句点までの文字列を意味する）のみを抜き出し，そこからさらに名詞のみを抽出して TF-IDF モジュールに入力するようにした．

文章から名詞のみを抽出するためには MeCab の形態素解析機能を用いた．よって，各クラスタを代表する重要単語（ユーザに提示する選択肢）は常に名詞となる．

一方，口コミデータセットからネガティブでない文のみを抽出するためには日本語評価極性辞書 [10] を使用した．ここで評価とは「〜が良かった」「〜が悪かった」というような個人の感想である．たとえば，ある観光地において「ここは景色が悪い.」というようなネガティブな口コミが多かった場合，この文を TF-IDF に入力すると，「景色」が重要語として選ばれる可能性があるが，ユーザは景色が良いのかと誤解するおそれがある．よってネガティブな文から抽出した単語をクラスタの代表単語として選定することは望ましくないと考えた，ネガティブと判定された文から得られた単語は TF-IDF に入力しないようにした．

日本語評価極性辞書には評価に用いられる単語とその極性（ネガティブ語/ポジティブ語）の 5,278 組が登録されているが，ネガティブと判定された文から得られた単語は TF-IDF に入力しないようにした．たとえば日本語評価極性辞書に登録されているネガティブ評価語には，悪い，低い，不足だ，下級だ，ひどい，などが含まれている．

今回使用した観光スポットの口コミ文の総数は 509 文であったが，ネガティブ判定された文がその中に 22 文含まれていた．例えば，「パンダコーナーは凄く混んでいましたので，諦めて他をまわった方が良いと思います (上野動物園)」，「ミッキーマウスの家には長蛇の列があまりに長すぎて断念しました (東京ディズニーリゾート)」，「電車好きでない人にはわかりませんが（鉄道博物館）」が含まれていた．最後の文については「好き」というポジティブ語が「でない」によって否定されているためネガティブ判定された．

(4) 選択クラスタの再帰的再分割：(3) で表示された選択肢のいずれかがユーザによって選ばれると，その選択肢に対応したクラスタが選ばれたとみなす．選ばれたクラスタは，(2) に戻って K 平均法 (K-Means) によって所定個のクラスタに再分割され，さらに，各クラスタに対して同様に (3)，(4) が施される．そして最終的にユーザが選択したクラスタ内に 1 つしか旅行先が含

まれなくなった時点で，その旅行先を推薦旅行先として Web ブラウザに表示するようにした．
なお，再帰的処理の途中で同じ名詞がクラスタを代表する重要名詞として繰り返し表示される可能性があるが，すでにユーザによって選択された名詞であることから繰り返し選択させるのは適当ではないと考えた．そこで，最も TF-IDF 値が大きい名詞が表示済み名詞であった場合は，2 番目に TF-IDF 値が大きい名詞（それも表示済みであれば 3 番目に TF-IDF 値が大きい名詞）を選択肢として表示するようにした．

4.2 動作説明

極めて小さいデータセットを用いて実際の動作を説明する．データセット内の旅行先は場所 A,B,C,D の 4 箇所とし，それぞれの旅行先に口コミが 1 件ずつ登録されていたとする．全口コミ文は以下の通りである．

場所 A	山 と 木 が 高い
場所 B	湖 と 花 が きれい
場所 C	川 と 湖 が きれい
場所 D	山 と ビル が 高い

これらは MeCab によって分かち書きされたものである．これらの 4 つの口コミ文は (1) において Doc2Vec によってそれぞれベクトル化される．

この例では，(2) において 2 個のクラスタに分割することとする．(1) で作成された 4 つの口コミ文のベクトルを K-Means モジュールに入力した結果，場所 A と場所 D がクラスタ 0 に，場所 B と場所 C がクラスタ 1 に分割された．大まかに「高い」に関係する場所と「きれい」に関係する場所に分かれる結果となった．

(3) では，(2) で作成されたクラスタ 0 とクラスタ 1 から TF-IDF 法を用いた重要語抽出を行う．この処理のために，以下のように場所 A と場所 D の口コミをつなげた 2 文がクラスタ 0 の口コミ文章となり，場所 B と場所 C の口コミをつなげた 2 文がクラスタ 1 の口コミ文章となる．

クラスタ 0 (場所 A,D)	山 と 木 が 高い．山 と ビル が 高い
クラスタ 1 (場所 B,C)	湖 と 花 が きれい．川 と 湖 が きれい

さらに，それぞれの口コミにネガティブな文があれば日本語評価極性辞書を用いて除去し，その後 MeCab で名詞を抜き出す．この例ではネガティブな文はないので，名詞抜き出し処理のみが実行される．すなわち，以下の口コミが TF-IDF に入力されることとなる．

クラスタ 0 (場所 A,D)	山 木 山 ビル
クラスタ 1 (場所 B,C)	湖 花 川 湖

結果として，クラスタ 0 では「山」が最も TF-IDF 値が大きく，また，クラスタ 1 では「湖」が最も TF-IDF 値が大きくなる．よってシステムは「山」と「湖」の選択肢を示し，どちらがより好みかをユーザに質問することとなる．

(4) においてユーザが「山」を選択したと想定する．この場合，クラスタ 0 が選ばれたとみなされることから，クラスタ 0 に入っている場所 A と場所 D の口コミ文章だけを使って (2) のクラスタリング処理を再実行することとなる．すなわち，以下の 2 つの口コミ文のベクトルを K-Means モジュールに入力して 2 個のクラスタに分割する．結果，場所 A がクラスタ 0 に，場所 D がクラスタ 1 に分割される．

場所 A	山 と 木 が 高い
場所 D	山 と ビル が 高い

(3) の重要語抽出では，クラスタ 0 からは「山」と「木」が，クラスタ 1 からは「山」と「ビル」が抽出されるが，「山」はユーザが既に選択肢として表示済みの単語であるので選択肢から除外される．よって，システムは「木」と「ビル」の選択肢を示し，どちらがより好みかをユーザに質問することとなる．

(4) でユーザが「木」を選択するとクラスタ 0 が選ばれたことになるが，クラスタ 0 内には場所 A の 1 箇所しか旅行先が含まれないため推薦旅行先確定となり，「あなたにオススメする旅行先は”場所 A”です．」と表示して終了となる．

4travel から取得した関東地方の有名観光スポット 70 箇所のデータセットを用いた場合，例えば，一度に表示される選択肢が 3 個の場合は約 4 回，選択肢が 4 個の場合は約 3 回，選択肢が 7 個の場合は約 2 回選択操作をすればおすすめ旅行先が表示される．データセットに含まれる旅行先数を N，一度に表示される選択肢数を a とすると，理論上，ユーザが選択する平均回数は $\log_a N$ である．

5. 評価

試作した旅ゲーターを Web サーバ上に構築し，大学生 8 名に使用させる実験を行った．データベースには 4travel から取得した関東地方の観光スポット 70 地点，口コミ文の総数 509 文が入っている状態である．被験者実験を行うに際し，被験者にはシステムの使い方について以下の文面で説明した．

『表示されている複数の選択肢（ボタン）から好きなものを一つ選んでください．選択肢数は自分で変更できます．「あなたにおすすめの旅行先は：」に情報が表示されたら終わりです．最初に戻ります．』

旅ゲーターを使用した場合と使用しなかった場合を比べるため，比較対象として観光スポット 70 地点の全口コミ文が入力されている一覧表（図 4 のエクセル文書）を用意した．そして旅行先を探すためにかかった時間（秒数）を

計測してもらった．システムを使用する場合は，選択肢数を 3,5,7,9 と変えて使用するよう指示すると共に，最も使いやすいと感じた選択肢数はいくつであったかを尋ねた．一覧表を使用する場合，どれだけ口コミ文を読むかは被験者の自由とし，全口コミ文を読むことは強制しなかった．

結果を述べる．旅行先を探すためにかかった時間については，システムを用いた場合が平均 18 秒，一覧表を用いた場合が平均 149 秒となり，システムを用いた場合は約 1/8 に必要時間が短縮される結果となった．なお，選択肢数毎の必要時間は，選択肢数が 3,5,7,9 のとき，それぞれ平均 22 秒，17 秒，17 秒，16 秒であった．最も使いやすいと感じた選択肢数については，選択肢数 3 が 1 名，5 が 5 名，7 が 1 名，9 が 1 名であった．

なお，被験者の中の 2 名に一覧表の口コミ文をすべて読むのにかかる時間を計測するよう依頼したところ，平均 695 秒かかる結果となった．一覧表を用いた場合の平均 149 秒は 695 秒の約 21 %であるが，これを鑑みると，口コミ文の 1/5 程を読んで旅行先を決めた（決めざるを得なかった）被験者が多かったと推測できる．また本実験では口コミサイトを検索した結果を一覧表にまとめたものを被験者に見せるようにしたが，被験者自ら旅行サイトを繰り返し検索して口コミを読むようにさせた場合には，一覧表を読むよりさらに多くの時間が必要になると予想される．

次に，旅ゲーターを使用した場合と使用しなかった場合のそれぞれについて，以下の 5 段階アンケート（1:あてはまらない ～ 5:あてはまる）に回答するように依頼した．

5 段階アンケートの結果を表 1 に示す．

表 1　アンケート調査結果

質問項目	旅ゲーター	一覧表
手軽に行ってみたい旅行先が見つかった	4.4	1.9
楽しく旅行先を見つけることができた	4.0	1.8
行ってみたい旅行先が見つかった	4.3	2.9

ウィルコクソンの符号付き順位和検定の結果，「手軽に行ってみたい旅行先が見つかった」と「楽しく旅行先を見つけることができた」の項目については有意水準 両側 5 %で有意差が認められた．「行ってみたい旅行先が見つかった」については統計的な有意差は認められなかったが，「従来の約 1/8 の時間しかかからなかったにも関わらず，一覧表を読んで見つけた旅行先に比べてシステムが提案した旅行先は劣っていなかった」ことを示す結果となっており，一定の評価がされたと考えている．

次に被験者からヒアリングした感想について述べる．システム使用時の感想を表 2 に，一覧表使用時の感想を表 3 に示す．

6. 考察

実験結果について考察する．

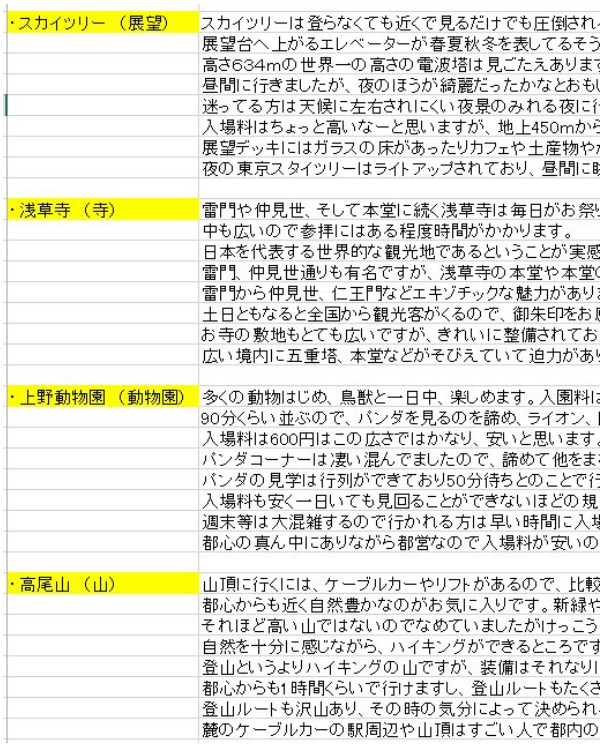

図 4　全口コミ文が入力されている一覧表

表 2　システム使用時の感想

ユーザの好みで旅行先を勧めてくれるから役に立ちます
選択してからすぐに結果が出て急いでいるときにいいと思った
自分の知らない所が出てきて，行ってみたくなった
選択肢の数を自分で変えることができるので，いろいろな場面に合わせて探しやすいと思った
選択肢 3 個の時だと好きな単語がない時があったが選択肢 5 個なら最低 1 つはあった
選択肢が多いと選ぶのが億劫になるので 5 がちょうど良いと感じた
行ったことがある場所が出てきてしまうことがあったので，幾つか推薦してもらいたかった

表 3　一覧表使用時の感想

長すぎるので読むのに疲れました．途中で読むのをやめました
多くて探す気がなくなる
多くの情報があり，分かりにくかった
ひとつずつ読んでいくのが大変だと感じました
行きたいと思う所が何箇所か見つかったとしても，結局どこが 1 番いいのか選ぶのに時間がかかった

6.1 適切な選択肢数

最も使いやすいと感じた選択肢数のアンケート結果，および，ユーザの「選択肢 3 個の時だと好きな単語がない時があったが選択肢 5 個なら最低 1 つはあった」「選択肢が多いと選ぶのが億劫になるので 5 がちょうど良いと感じた」という感想から，選択肢数は 5 が最適であったことがわかった．よってシステムのデフォルト選択肢数は 5 に設定するようにした．

選択肢数について考察する．Iyengar(アイエンガー) [18] は，選択肢数と満足度の関係について複数の実験を行い，選択肢数の増加が選択結果の満足度を低下させ，結果として選択行動そのものを放棄させる場合があることを示している．スーパーマーケットのジャム試食コーナーにおいて，6 種類のジャムを陳列した場合と，24 種類のジャムを陳列した場合とを比較した実験では，6 種類の場合は 40 %の人が試食コーナーで立ち止まってその中の 30 %の人が購入した一方で，24 種類の場合は 60 %人が立ち止まったもののその中の 3 %の人しか購入しなかった．結果としてジャムの購入件数は 6 種類の場合が，24 種類の場合の約 6 倍であったと報告している．

また大学生を対象にチョコレートの選択行動を観察した実験では，30 種類から最も好みのものを選択した場合と，6 種類から最も好みのものを選択した場合とを比較した場合に，6 種類から選んだ方が，選択したチョコレートに対する満足度が高く，選択したチョコレートをより美味しいと感じた人が多かったと報告している．Iyengar は選択肢数の増加によって選択行動が放棄されたり，選択結果満足度が低下する現象を「選択のオーバーロード」と呼んでいる．そして，人が自信を持って選べ，選んだ結果について満足度が高いのは選択肢数が 5〜9（7 ± 2）の場合が多いと述べている．

加えて Schwartz(シュワルツ)[19] は，選択肢が多ければ多いほど不幸を感じやすくなってしまう人間の心理作用のことを「選択のパラドックス」と呼んでいる．選択肢が多いと，苦労して多くの情報を比較して 1 つを選んだとしても，「他の選択もあったのでは」という迷いが大きくなってしまい，正しい選択をしたかどうかに自信が持てなくなり満足感が低下する．そのため，多くの中から選択する度に不幸を感じてしまうと述べている．

旅ゲーターにおいて選択肢数で「全て」を選択すると，システムに登録されている旅行先数の選択肢を表示するようになっており，旅行先候補が 70 箇所の場合には 70 個の選択肢が表示される．この場合，各旅行先の口コミ文章の中から TF-IDF 値が最大の名詞が 1 つずつ抽出され選択肢として示される．ただし前述のとおり，選択肢数の増加が選択結果の満足度を低下させる可能性が高く，選択肢数で「全て」が選べるのは補助的な機能との位置づけである．

6.2 代替案表示機能の追加

「行ったことがある場所が出てきてしまうことがあったので，幾つか推薦してもらいたかった」という感想については，対応が必要であると考え，実験実施後にシステムを改良した．具体的には，図 3 の「あなたの好み：」に表示されている「桜，子供」のような，ユーザが選択したすべての名詞を使い，これらの名詞群に近い旅行先をデータセットから改めて検索して代案として表示するようにした．

この機能を実装するための方法を 2 つ試した．前述した通り，全旅行先の全口コミ文章は gensim の Doc2Vec ライブラリによってベクトル化されているが，1 つ目の方法は Doc2Vec ライブラリが提供する infer_vector() メソッドを用いる方法である．infer_vector() の引数に「’山’,’ と’,’ 木’,’ が’,’ 高い’」のような分かち書きした任意文章を入力すれば，全口コミ文章から類似した文章を検索することができる．そこで，この infer_vector() の引数に「’ 桜’,’ 子供’」のようなユーザが選択した名詞群を入力するように実装した．

2 つ目の方法は，全旅行先の全口コミ文章の名詞を TF-IDF を用いてベクトル化し，「’ 桜’,’ 子供’」などのユーザが選択した名詞群が持つベクトルとのコサイン類似度を計算して類似文章を検索する方法である．具体的には gensim ライブラリの cosine_similarity() メソッドを用いることで実装した．

2 つの方法を使用して出力された代替案のどちらがより適切か検討したところ，結果としては大抵の場合において TF-IDF を用いた場合の方がより適切であった．Doc2Vec を用いた場合はユーザが選択した名詞が 1 つも含まれない旅行先が検索されるケースも少なからずあった．

推測される理由であるが，Doc2Vec ライブラリが提供する infer_vector() メソッドは，そもそも任意の自然文章を入力してそれに近い文章を検索するためのものである．Doc2Vec ライブラリによって作成された口コミ文章のベクトルは単語の共起関係を保持しているが，ユーザが選択した名詞群には自然言語が持つような共起関係が存在しない．この両者を比較するようにしたために，Doc2Vec の実力が発揮されなかったと思われる．

一方，全口コミ文章の名詞を TF-IDF を用いてベクトル化する場合は単語の出現回数のみを用いるため，作成されたベクトルは単語の共起関係を保持していない．これが，ユーザが選択した名詞群に共起関係が存在しないこととマッチし，Doc2Vec よりも適切な文章を検索する結果となった可能性がある．よって TF-IDF による方法を採用することとした．

図 5 に，追加機能を実装したシステム使用時のスナップショットを示す．案 A 以外が代替案表示機能によって作成された案である．TF-IDF のコサイン類似度が高い順に案 B，案 C，案 D... である．案 A を含めて幾つの案を表示す

るかをユーザが「推薦数」で選べるようにした．

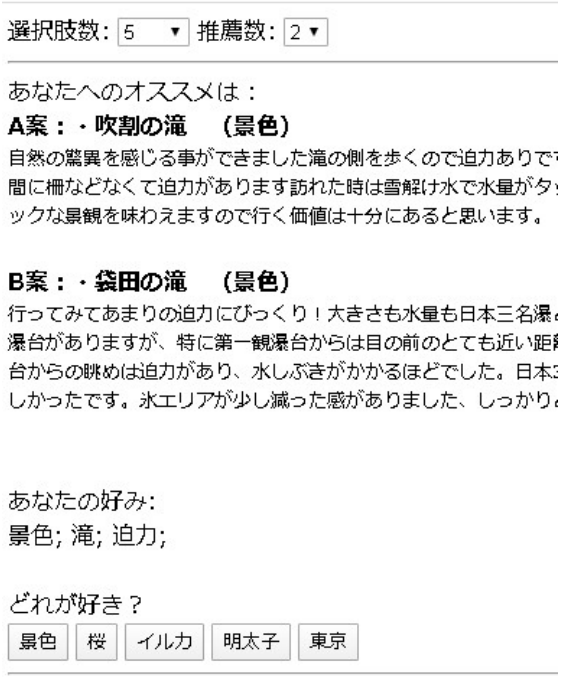

図 5　代替案表示機能の追加

7. まとめ

旅行計画を立てる際に多くの旅行情報を閲覧しなくても手軽に旅行先を選ぶことができるようにするためのシステム「旅ゲーター」を提案した．口コミ解析と好み診断に基づいて旅行先推薦する機能が特徴である．

評価実験の結果から，手軽に行ってみたい旅行先が見つかったかどうか，楽しく旅行先を見つけることができたかどうか，行ってみたい旅行先がみつかったかどうか，の全ての評価項目において，口コミ文章を読んで旅行先を見つけるという従来の方法より，旅ゲーターを利用した方が良い結果となった．

また，ユーザから「行ったことがある場所が出てきてしまうことがあったので，幾つか推薦してもらいたかった」という意見があり，それに対応して代替案表示機能を追加実装した．この機能の評価，および，より適切な実装方法の検討については今後の課題としたい．

なお本研究は JSPS 科研費 16K00506 の助成を受けたものです．

参考文献

[1] 旅工房 –最新！旅行意識調査– 7 割が旅行の「計画疲れ」に悩む！ https://www.tabikobo.com/company/news/press/2018/06/180605 (2018)
[2] 大山，竹川，平田：レビュー文を考慮したゲーム推薦システムの実現に向けた単語の類似度調整の取り組み，情報処理学会エンタテインメントコンピューティングシンポジウム 2017 論文集，,
[3] 梅本，豊田，大原：料理レシピの分散表現を用いた代替食材の発見手法，情報処理学会 行動変容と社会システム Vol.03 (2018)
[4] 谷本, 太田：評価属性を考慮した評判情報の可視化, 情報処理学会 DBS 研究報告, Vol.151, No.4, pp.1-8 (2010)
[5] 平山，湯本，新居，佐藤：語の共起と極性に基づく商品レビュー閲覧支援システム, 情報処理学会 DBS 研究報告, Vol.155, No.3, pp.1-9 (2012)
[6] 中野，湯本，新居，佐藤：商品レビュー要約のための属性-意見ペア抽出，情報処理学会 DBS 研究報告，Vol.160，No.15,pp.1-7 (2014)
[7] 紀本，伊藤，宗森：評価表現に着目した映画レビューからの評価情報抽出，情報処理学会 DCC 研究会報告 Vol.21, No.43, pp.1-8 (2019)
[8] Erkan G., Radev, D.R. : LexRank - Graph-based Lexical Centrality as Salience in Text Summarization, http://www.cs.cmu.edu/afs/cs/project/jair/pub/volume22/erkan04a-html/erkan04a.html (2004).
[9] SlothLib，Web 検索研究支援プログラミングライブラリ，日本語ストップワード，http://svn.sourceforge.jp/svnroot/slothlib/CSharp/Version1/SlothLib/NLP/Filter/StopWord/word/Japanese.txt (2019) pp. 223-227 (2017)
[10] 小林，乾，松本，立石，福島：意見抽出のための評価表現の収集，日本語評価極性辞書（用言編），自然言語処理，Vol.12, No.3, pp.203-222 (2005)
[11] 工藤：MeCab - Yet Another Part-of-Speech and Morphological Analyzer, https://taku910.github.io/mecab/ (2019)
[12] mecab-ipadic-NEologd : Neologism dictionary for MeCab, https://github.com/neologd/mecab-ipadic-NEologd/blob/master/README.ja.md (2019)
[13] Lau,J.H. and Baldwin,T.: An Empirical Evaluation of Doc2Vec with Practical Insights into Document Embedding Generation, https://arxiv.org/abs/1607.05368 (2016)
[14] models.doc2vec - Doc2vec paragraph embeddings, https://radimrehurek.com/gensim/models/doc2vec.html (2019)
[15] models.word2vec - Word2vec embeddings, https://radimrehurek.com/gensim/models/word2vec.html (2019)
[16] gensim, Free Python Library, https://radimrehurek.com/gensim/ (2020)
[17] https://www.python.org/ (2019)
[18] Iyengar, S.：選択の科学、コロンビア大学ビジネススクール特別講義、文春文庫 (2010).
[19] Schwartz, B.：選択のパラドックスについて，https://www.ted.com/talks/barry_schwartz_on_the_paradox_of_choice, TED Global (July 2005).