一般社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
LOIS2015-14 (2015-07)

潜在的興味に基づく観光地推薦システムの試作

開地 亮太† 檜垣 泰彦‡

†‡ 千葉大学大学院工学研究科〒263-8522 千葉県千葉市稲毛区弥生町 1-33
E-mail: †afpa3246@chiba-u.jp, ‡higaki.yashiko@faculty.chiba-u.jp

あらまし 本研究では Web 上にある膨大な観光情報をテキストデータとして取得し，単語の分散表現を用いてユーザーの潜在的興味を発見し，その潜在的興味に基づいた観光地を推薦するシステムを試作した．Wikipedia の見出し語を観光関連度で分類することで観光地データベースを作成し，これを用いて観光情報を収集した．単語の分散表現は収集した観光情報を word2vec で学習することによって獲得した．作成した観光地推薦システムを iOS アプリケーションとして実装した．アプリケーションには推薦，お気に入り登録，履歴確認，自動推薦の機能を追加した．アプリケーションにユーザーの使用履歴を継続的に収集する機能を実装し，ユーザーの使用履歴から単語ベクトルの更新を行うシステムを追加した．今後の課題は提案した観光地推薦システムの評価を行うこと，ユーザーの入力に表記のゆれが発生した際に起きる問題の解決を行うことである．

キーワード word2vec，観光地推薦システム，大規模文書集合，単語ベクトル，潜在的興味

Recommendation system of tourist spots using large amounts of tourism information

Ryota KAICHI† Yasuhiko HIGAKI‡

†‡ Graduate School of Engineering, Chiba University 1-33 Yayoi-cho, Inage-ku, Chiba-shi, 263-8522 Japan
E-mail: †afpa3246@chiba-u.jp, ‡higaki.yasuhiko@faculty.chiba-u.jp

Abstract The authors developed recommendation system of tourist spots that is based on the latent interest. Latent interest was acquired from word vectors that made by learning from text data of tourism information on the Web. A tourist spots database was made by classifying Wikipedia keywords in tourism-related degree. And tourism information was collected by this database. Word vectors were acquired by learning tourism information using word2vec. The authors have implemented the recommendation system of tourist spots as iOS application. Some of the features, such as recommendation, bookmark, history check and automatic recommendation were added to the application. And the authors have added a system for updating the word vector from the user's usage history. To evaluate recommendation system of tourist spots and to solve the problem of spelling are future challenges.

Keywords word2vec，Recommendation system of tourist spots，Large amounts of information，Latent interest

1．はじめに

近年，スマートフォンや PC の普及によりブログや SNS から大量の情報の取得や発信が出来るようになった．特に観光の分野では観光者のブログへの投稿や，全国各地の観光施設や団体が発信している情報など多数の情報を取得できる．しかしそれらの情報は膨大であり，ユーザーの嗜好にあった情報を見つけ出すことは困難である．一方，観光産業は世界的にも大きな経済部門であり，日本では富士山が世界遺産に登録されるなど経済効果の面で追い風が吹いており，観光における推薦技術は社会的ニーズが高いと考えられている．

そのため観光地推薦に関連する研究は盛んに行われている．ユーザーのお気に入りの観光地に類似した観光地を推薦する研究[1]や位置情報から観光ルートを推薦する研究[2]はあるものの，興味の足し引きによって潜在的興味を発見し，潜在的興味に沿った推薦を行う研究はまだ行われていない．潜在的興味に関する研究は[3][4]など行われており，注目を集めている．ユーザーが普段から意識している既知の興味に基づく推薦だけでなく，ユーザーが今まで意識することがなかった潜在的興味に基づく推薦が求められている．

そこで本研究では Web 上にある膨大な観光情報をテキストデータとして取得し，単語の分散表現を用いてユーザーの潜在的興味を発見し，潜在的興味に基づいた観光地を推薦するシステムを試作した．

2．目的

本研究の目的は，観光地にマンネリを感じ，自分で

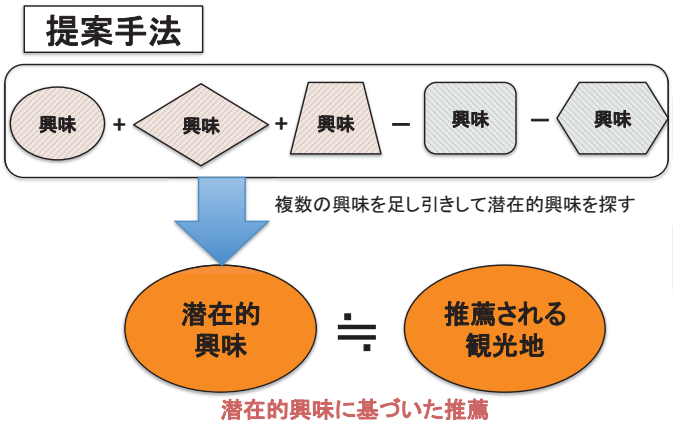

図 1 提案手法

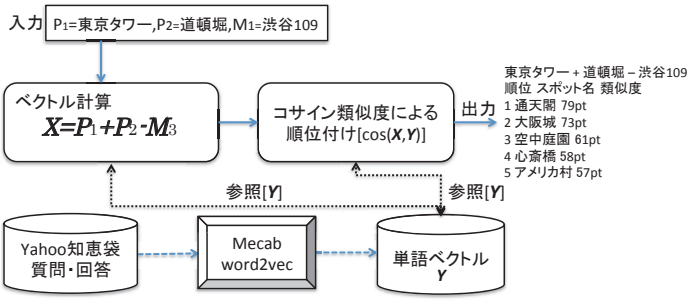

図 2 システムの流れ

は思いつかないような新たな観光地を求めているユーザーに対し，観光地を推薦することである．また，ユーザーの嗜好は千差万別であると考えられる点,今後，観光地推薦システムの評価を行う点を考え，以下の要件を満たす観光地推薦システムを構築する．

(1) 潜在的興味が発見できること
(2) 観光地を幅広く網羅していること
(3) 新しくできた観光地の情報も取り扱うこと
(4) ユーザーの使用履歴を継続的に取得できること
(5) 推薦された観光地についてユーザーが興味を持ったか判断できること

(1)については 3 節の方法により，(2)については 3 節と 4 節の方法，(3)(4)(5)については 4 節の方法により要件を満たす．

3. 潜在的興味の発見方法と推薦手順

3.1.潜在的興味

本論文では潜在的興味をユーザーの好きな物や嫌いな物の特性を足し引きしたものであると設定する．本システムに当てはめると好きな観光地や嫌いな観光地の特性を足し引きしたものが観光における潜在的興味である．足し引きによって得た潜在的興味に基づいた推薦を行うことで，ユーザーの嗜好を捉えつつユーザーが思いつかないような観光地を推薦できる．本システムで提案する潜在的興味発見までの概略図を図 1 に示す．図 1 に示した手法を実現するために作成した推薦システムの流れを図 2 に示す．図 2 に示した推薦システムについて順次述べる．

3.2.潜在的興味の発見方法

3.1.節で述べた観光地の特性を足し引きするため，単語の意味を計算できる形にする必要がある．自然言語処理研究において単語が持つ意味情報をベクトルによって表現する単語ベクトル空間モデル[5]が広く用いられてきた．このモデルは似た文脈に出現する単語は似た意味を持つという仮説である Distributional Semantic Models[6]に基づいている．近年ではニューラルネットワークによる学習モデルである Skip-gram モデル[7]が Mikolov らによって新たに提案され，従来の手法に比べ飛躍的に精度が向上した．また Skip-gram モデルを用いることにより単語の足し引きが可能となり'Paris' - 'France' + 'Italy'の計算を行うと'Rome'になるという報告が Mikolov らによってされている．本システムでは観光地の特性を足し引きするため，観光地の単語ベクトルを作成し，足し引きを行う手法を提案する．単語ベクトルは単語を固定長のベクトルとして表現する手法である．単語を空間上の一点に対応することができるため，単語が持つ意味の近さを数値で表現できる．つまり，単語の意味計算を行うことが可能になる．単語ベクトルを大規模文書集合(コーパス)から学習，作成するツールとして Mikolov らの手法を実装した word2vec[1]を使用した．

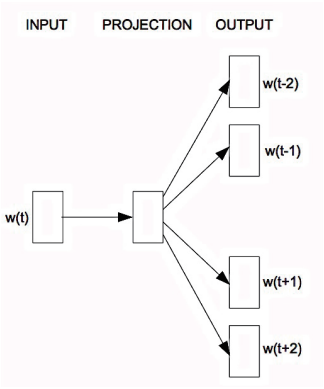

図 3 Skip-gram

3.3.word2vec

word2vec とはニューラルネットワークを用いた単語の分散表現(単語ベクトル)を作成する手法である．Mikolov らは Skip-gram モデルでの学習により単語ベクトルを作成する手法を提案した．Skip-gram モデルの概略図を図 3 に示す．入力コーパスの 1 文$(w_1 \dots w_r)$が与えられると Skip-gram モデルは次式を最大化するような単語ベクトルを求める．

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\le j\le c, j\neq 0} \log p(w_{t+j}|w_t) \qquad (1)$$

ここで c は中心単語w_tに対してどの距離までの単語を文脈とするか設定する定数である．確率$p(w_{t+j}|w_t)$は log-bilinear モデル[8]と呼ばれ以下のように定式化される．

[1] https://code.google.com/p/word2vec

$$p(w_{t+j}|w_t) = \frac{\exp(v_{w_t} \cdot u_{w_{t+j}})}{\sum_{w^T} \exp(v_{w_t} \cdot u_{w^T})} \quad (2)$$

確率$p(w_{t+j}|w_t)$は単語w_tから周辺の単語w_{t+j}を予測するものである．したがって周辺の単語の分布が似ていれば単語ベクトルは似た値をとる，すなわち単語ベクトルの値が近いほど単語の意味も近くなると言える．word2vec によって単語を分散表現に変換することにより単語を意味空間上の１点に対応することができ，単語に対する意味的な計算が可能になった．入力コーパスを観光に関係あるものにし，観光地の単語ベクトルを作成することで観光地同士の意味的な計算ができると考えられる．

3.4.観光地データベースの作成

word2vec はコーパスの量が大きいほど良い精度になる報告がされている[7]．つまり本論文においては観光地に関係のある文書を多く取得することが望ましい．本論文では Yahoo!知恵袋[2]の「地域，旅行，お出かけ」>「国内」カテゴリを対象に観光地を検索クエリとし，ヒットした質問・ベストアンサーを取得する．そのため初めに観光地を集めた観光地データベースの作成を行う．本論文では「観光に関係のある場所」を観光地と設定する．しかし「観光に関係のある場所」は人の主観によって変化するため，客観的な判断で定める必要がある．客観的な判断で観光地か否かの判定をするため Yahoo!知恵袋のカテゴリを利用した．図 4 は Yahoo!知恵袋のカテゴリ構造を大まかに表したものである．カテゴリ分けがされていることから，あるキーワードが他のカテゴリに比べ観光カテゴリに多く存在していたとすると，このキーワードは観光に関係の深いキーワードであると言える．この特性を用いることで観光地の判定，取得を行う．観光地データベース作成手順は以下の通りとなる．

(1) 日本語版 Wikipedia[3]の見出し語約 100 万件を記述したテキストデータを取得し，英字・数字・記号などのノイズが含まれるものを不用語として取り除く

(2) (1)で選別した見出し語１つひとつを Yahoo!知恵袋 API[4]を使用して２つの方法で検索し，総ヒット件数を見出し語ごとに保存する．

a) 「地域，旅行，お出かけ」>「国内」カテゴリで検索

b) カテゴリを指定せずに検索

ここで見出し語kにおける(a)での総ヒット件数をJ_k，(b)での総ヒット件数をA_kとする．

(3) Wikipedia の見出し語が観光にどれくらい関連

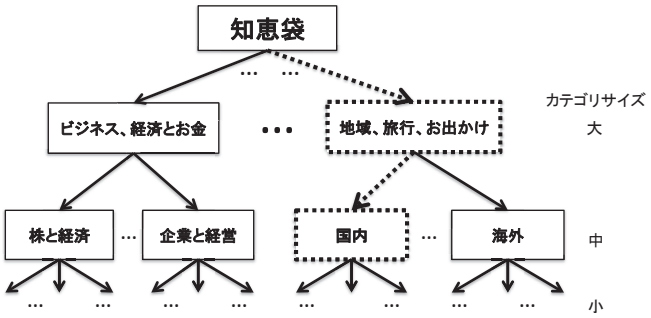

図 4 知恵袋のカテゴリ構造

表 1 ヒット件数

見出し語k	ヒット件数J_k	ヒット件数A_k	観光関連度S_k
東京スカイツリー	3,482	6,079	0.57
黒部ダム	1,319	1,674	0.79
青森ねぶた祭り	73	122	0.59
隅田川花火大会	591	734	0.81
ラーメン	12,016	124,846	0.1
東京ドーム	3,085	59,877	0.05

しているか表す度合いである観光関連度S_kを

$$S_k = \frac{J_k}{A_k} \quad (3)$$

と定義する．(ただし$A_k \geq 3$)

(4) $S_k \geq 0.5$ のものを観光地と判断して観光地データベースに登録する．

これらにより 9034 件の観光地を登録した観光地データベースが完成した．Wikipedia の見出し語ごとのヒット件数一部を表 1 に示す．

表 1 よりラーメンや東京ドームといった見出し語は他のカテゴリでも多く出現するため観光関連度は低い値となることがわかる．また，青森ねぶた祭りや隅田川花火大会など年に一度だけ行われるようなイベントも観光地として登録されることが分かった．これらのイベントは開催期間に観光客が大勢訪れ一時的な観光地となる．この手法を用いることで数日だけ現れる観光地の判別も可能であり，より多様的な観光地データベースの作成が可能である．

3.5.コーパスの作成

コーパスを作成する手段として 3.3.節で作成した観光地データベースの情報を活用する．Yahoo!知恵袋 API を使用し，Yahoo!知恵袋の「地域，旅行，お出かけ」>「国内」カテゴリーを対象に観光地データベースに登録された観光地を検索クエリとし，ヒットした質問・ベストアンサーを取得する．取得した文書は形態素解析エンジン Mecab[5]を使用し文書を品詞ごとに

[2] http://chiebukuro.yahoo.co.jp
[3] http://ja.wikipedia.org/wiki
[4] http://developer.yahoo.co.jp/webapi/chiebukuro/chiebukuro/v1/questionsearch.html
[5] https://code.google.com/p/mecab/

表 2 オプション

train:学習対象のテキストデータ	2.05GB
size:次元数	200 次元
windows:ウィンドサイズ	5 単語
hs:階層的ソフトマックス	1(使う)
sample:単語を無視する際の頻度の閾値	10^{-3}

分解する．ここでは Mecab の辞書には観光地データベースを登録した Wikipedia の見出し語 9034 件を追加した．

3.6.学習設定

単語ベクトル取得には 3.2.節での手法を用いる．本研究では出力する単語ベクトルの次元数は 200 次元に設定した．その際 word2vec は表 2 に示すオプションで実行した．学習により単語ベクトルを取得し単語ベクトルデータベースに保存した．

3.7.コサイン類似度

本論文ではユーザーに好きな観光地（P_n）嫌いな観光地(M_n)を n 個入力してもらうことを想定している．3.5.節で作成した単語ベクトルをユーザーの入力に適応するとそれぞれの単語ベクトル$\boldsymbol{P_n}, \boldsymbol{M_n}$は，

$$\boldsymbol{P}_n = \{p_{n_1}, p_{n_2}, \dots, p_{n_{200}}\} \quad (4)$$

$$\boldsymbol{M}_n = \{m_{n1}, m_{n2}, \dots, m_{n200}\} \quad (5)$$

と表せる．よって入力ベクトル $\boldsymbol{X}$ は，

$$\boldsymbol{X} = \boldsymbol{P}_1 + \boldsymbol{P}_2 \dots + \boldsymbol{P}_n - \boldsymbol{M}_1 - \boldsymbol{M}_2 \dots - \boldsymbol{M}_n \quad (6)$$

と表現できる．

また 2.5.節で作成した全単語ベクトル m 個の単語ベクトル$\boldsymbol{Y}_m$は，

$$\boldsymbol{Y}_m = \{y_{m_1}, y_{m_2}, \dots, y_{m_{200}}\} \quad (7)$$

と表せる．

本論文では類似度計算処理としてベクトル空間モデルのコサイン類似度の計算を行う．ベクトル $\boldsymbol{X}$, $\boldsymbol{Y}$ のコサイン類似度$\cos(\boldsymbol{X}, \boldsymbol{Y})$は，

$$\cos(\boldsymbol{X}, \boldsymbol{Y}) = \frac{\boldsymbol{X} \cdot \boldsymbol{Y}}{|\boldsymbol{X}||\boldsymbol{Y}|} \quad (8)$$

で計算される．この計算により観光地の類似度の評価を行う．

4. 観光地推薦システム実装

4.1.アプリ構造

本観光地推薦システムは 2 節の(2)(3)(4)(5)の要件を満たすため iOS アプリケーションとして開発を行った．開発を行ったアプリケーション構造 を図 5 に，観光地推薦画面を図 6 に示す．ユーザーは好きな観光地と嫌いな観光地をそれぞれ 3 件まで入力し「結果へ」ボタンを押すことで図 2 に示した流れにより観光地を発見，推薦を行う．推薦された観光地の横にあるチェックボックスをタップすると観光地がお気に入りのリストに保存され，「お気に入り」のメニューからいつでもお気に入りの観光地を確認することができる．また，お気に入りリストをタップすると図 6 に示すようにアプリケーションから直接観光地に関する情報を調べることができる．

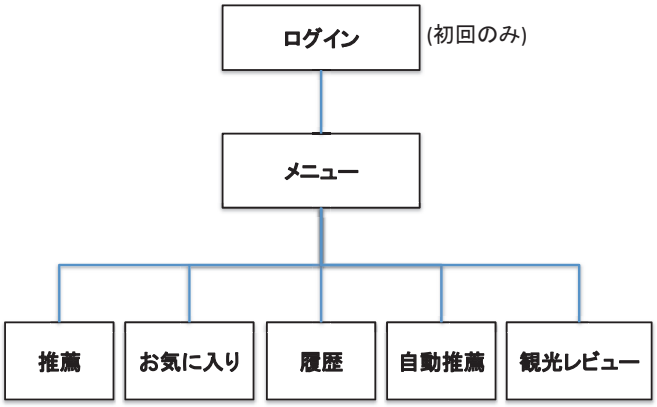

図 5 アプリ構造図

図 6 アプリ画面

またユーザーの使用履歴を収集する機能を実装した．主に収集するデータはユーザーid，推薦結果，お気に入りに登録した観光地，入力した好きな観光地，嫌いな観光地である．収集したデータは本観光地推薦システムの評価や自動推薦を行う機能で使用する．自動推薦とは，ユーザーが今まで入力した好きな観光地や嫌いな観光地，お気に入りに登録した観光地を入力クエリとし自動的に新しい観光地を推薦する機能である．ユーザーが直接観光地を入力する必要がないため，より潜在的な興味に基づいた観光地が推薦される．また収集したデータを使用して観光地推薦システムの自動更新を行う．

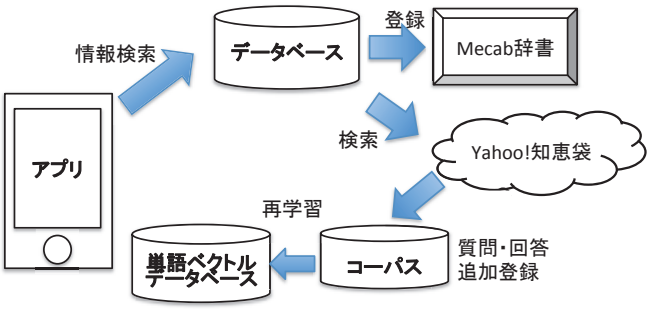

図 7 データベース自動更新

4.2.データベース自動更新

ユーザーが観光地推薦システムを使用する際，新しくできた観光地やマイナーな観光地の入力を行った場合など観光地データベースに登録されていない観光地の入力が行われる可能性がある．そこで本論文ではユーザーの入力データをもとに観光地データベースとコーパスを自動的に更新するシステムを構築した．観光地データベースとコーパスの自動更新の流れを図 7 に示す．観光地データベースとコーパスの自動更新の手順は以下のようである．

(1) ユーザーの入力データ取得
(2) 取得した入力データを観光地データベースに登録してあるか検索を行い，登録されていない場合登録，同時に Mecab 辞書に登録
(3) 新しく観光地データベースに登録された観光地を検索クエリとして Yahoo!知恵袋で検索を行い質問・ベストアンサーをコーパスに追加
(4) 毎日 2 時に学習を行い，単語ベクトルデータベースを更新

観光地データベースとコーパスの更新により新たに出来た観光地も取り扱え，さらにより幅広く観光地を網羅できると考えられる．

5. 考察

2 節で述べた要件を満たしているかについての評価を行った．

2 節(1)の要件について，潜在的興味が発見できたか判断するにはユーザーに実際に使用してもらう必要がある．本観光地推薦システムは試作でありアプリケーションの公開を行っていないため，まだ評価を行っていない．今後アプリケーションを公開し，ユーザーの使用履歴を用いることで要件(1)が満たされているかの確認を行うことが必要である．

2 節(2)の要件について観光地を幅広く網羅しているかの評価をアンケートにより行った．単語ベクトルはコーパスから作成され，コーパスは観光地データベースの観光地をもとに作成されることから観光地データベースにユーザーが必要としている観光地が正しく登録されているかについての確認を行った．アンケート対象は 32 名の学生とし，web アンケートによって以下の質問を行った．

表 3 よりアンケートからユーザーが思いついた観光地が登録されているかを表す適合率は 77.2%であり，登録されていなかった観光地は，仁田峠，国道 339 号線，靖国神社，富士山，花山公園，新潟などであった．なぜ登録されていない観光地が存在したのかについて表 4 で示す案件について検討を行った．

表 4 より登録されなかった観光地を分別すると

(a)Wikipedia の見出し語に存在しない観光地

(b)見出し語に存在するが観光関連度が 0.5 を超えなかった観光地

の二つに分類できる．

(a)に分類された仁田峠は観光関連度が高い値になっているが，見出し語が存在しなかった．これは，仁田峠は雲仙岳にある峠であり雲仙岳のページで説明されているため考えられる．また花山公園や新潟が見出し語に存在しなかったのは，Wikipedia の記事の作り方によるもので，Wikipedia のガイドラインで見出し語は正式名称で登録するよう述べている．よって基本的に愛称や略称，別名は見出し語としては存在しない．花山公園はつつじが岡公園の愛称で，新潟は新潟県及び新潟市の略称であるため見出し語には登録されていなかったと考えられる．これらのことから Wikipedia の見出し語を分類するだけでは全ての観光地を網羅することができないことが分かった．

(b)に分類された靖国神社，富士山は見出し語が存在するが観光関連度が低い値になったため，観光地データベースに登録されなかった．富士山や靖国神社は観光の面だけでなく，アウトドアや自然，歴史など多くの属性があり，様々なカテゴリで質問されるため観光

表 3 アンケート結果

質問	回答数	登録済件数	適合率
思いついた観光地を書いてください	57 件	44 件	77.2%

表 4 観光地の検討

観光地	見出し語の有無	観光関連度 $S_k = \frac{J_k}{A_k}$
仁田峠	無し	$0.83=176/211$
花山公園	無し	$1.0=3/3$
靖国神社	有り	$0.06=669/11258$
富士山	有り	$0.22=6708/30465$
新潟	無し	$0.13=19793/154730$

関連度が低い値になったと考えられる．多面的な属性を持つ観光地は観光関連度での分別では漏れが出てしまうことが分かった．

これらの問題点を解決するために本観光地推薦システムでは 4.2 節で述べたデータベース自動更新の機能を実装した．アンケートで得た観光地をアプリでの入力と仮定してデータベースの自動更新を行い，更新前と更新後の単語ベクトルの有無について調べた．結果を表 5 に示す．

表 5 に示す仁田峠のようにデータベース自動更新の機能により登録されていなかった観光地についての情報を追加することができた．花山公園は Yahoo!知恵袋でのヒット件数が少なかったため，文中での登場回数が少なく単語ベクトルの生成が行われなかったと考えられる．花山公園の正式名称であるつつじが岡公園の単語ベクトルは更新前にも存在しており，愛称を入力したことでヒット件数が減り単語ベクトルが生成されなかった．この点からユーザーが愛称を入力したときや，表記のゆれが発生したときに正式名称に変換することが必要であると考えられる．靖国神社，富士山，新潟は文中に比較的多く出現する単語であったため更新前からでも単語ベクトルが生成されていたと考えられる．

これらの例よりデータベース自動更新を行うことで，単語ベクトルの作成行えることが分かった．したがって，ユーザーに多く使ってもらうことにより，要件(2)の観光地を幅広く網羅することが期待される．

(4)(5)の要件については，今後アプリケーションを公開しユーザーに使用してもらうことにより，要件が満たされたかの確認が必要だ．

6. まとめ

本論文では，Web 上にある膨大な観光情報をテキストデータとして取得し，単語の分散表現を用いてユーザーの潜在的興味を発見し，潜在的興味に基づいた観光地を推薦するシステムを試作した．単語の分散表現は観光地データベースをもとに作成したコーパスを word2vec で学習することによって獲得した．また要件を満たすため，作成した観光地推薦システムを iOS アプリケーションに実装し，考察を行った．

表 5 単語ベクトルの有無

観光地	更新前	更新後
仁田峠	無し	あり
花山公園	無し	無し
靖国神社	あり	あり
富士山	あり	あり
新潟	あり	あり

考察により今後要件が満たされたかの確認を行うことが必要であることが分かった．また，本観光地推薦システムではユーザーが入力した単語をそのまま推薦システムの入力しているため，表記のゆれや略称や愛称といった複数の名前を持つ観光地が入力となる可能性がある．ユーザーが愛称を入力したときや，表記のゆれが発生したときに正式名称に変換することが必要であると考えられる．これらの問題の解決を行うことが今後の課題である．

文 献

[1] 上原 尚，嶋田和孝，遠藤 勉:Web 上に混在する観光情報を活用した観光地推薦システム:電子情報通信学会，言語理解とコミュニケーション研究会(NLC)，第 4 回集合知シンポジウム，NLC2012-35，pp. 13-18, 2012

[2] 中嶋勇人，新妻弘崇，太田学:位置情報付きツイートを利用した観光ルート推薦:情報処理学会研究報告，データベース・システム研究会報告 2013-DBS-158(28)，1-6，2013-11-19

[3] 近藤 光正，中辻 真，田中 明通：Wikipedia に基づく Web 閲覧履歴からの潜在的興味キーワード抽出：電子情報通信学会論文誌.D，情報・システム J96-D(5),1199-1211,2013-05-01

[4] 志甫谷 匠，中島 伸介，角谷 和俊：ユーザーの潜在的興味抽出とコミュニティの粒度推定に基づく情報推薦システム：全国大会講演論文集，第 72 回(データベースとメディア)，863-864，2010-03-08

[5] Peter Turney，Patrick Pantel : From frequency to meaning: Vector space models of semantics : Journal of Artificial Intelligence Research 37， 141–188，2010

[6] Stefan Evert，Marco Baroni， Alessandro Lenci : Distributional Semantic Models : NAACL HLT 2010 Tutorials，Los Angeles，1 June 2010

[7] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean : Efficient Estimation of Word Representations in Vector Space : In Proceedings of Workshop at ICLR，2013.

[8] Andriy Mnih， Yee Whye The : A fast and simple algorithm for training neural probabilistic language models : Pro- ceedings of the 29th International Conference on Ma- chine Learning，pp. 1751–1758 ，2012.