Vol.2019-GN-107 No.5
2019/3/18

口コミを利用した料理情報提供サービスの提案

市村 哲[1]　佐藤 美南[1]

概要：グルメサイトを利用する際，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．実際に食したであろう人たちの料理の感想は非常に有用であり，店舗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．しかしながら，グルメサイトに登録されている飲食店は非常に多く，かつ，飲食店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．この問題に着目し，著者らは，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．ユーザが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と，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．さらに，料理に関係する口コミ情報を優先的に表示し，かつ，長いレビュー文章をそのまま表示する代わりに，レビュー文章から抽出した料理に関する短い要約文を作成して表示する．

Restaurant Searching System Using Review

Satoshi Ichimura[1]　Minami Sato[1]

Abstract:
When using gourmet site, we often refer to impressions written by others (user reviews for short). The impression of the cooking of the people who actually would have eaten is very useful and it is possible to know a lot of information other than photographs and texts provided from the restaurant. However, there are a lot of restaurants registered in the gourmet site, and it is also common that a large amount of reviews is written on each restaurant. Focusing on this problem, we have developed a web service "Syoku-Tan", eating hunter in English, that can allow you to quickly find a restaurant where you can eat delicious food. When a user inputs his or her wishes and questions about what kind of dish they want to each in Japanese natural sentences, the system detects user reviews similar to the user's input sentence, and find restaurants suitable to the user. Furthermore, instead of displaying a long review sentence, the system creates a short summary extracted from the long review sentence, and displays it on the screen.

1. はじめに

インターネットで食事に関する情報を収集したり発信したりすることが一般的になっている．たとえば飲食店を検索することができるサイト（通称，グルメサイト）では，ユーザはまず場所や料理ジャンルなどの条件を入力して料理店を検索するが，それと同時に検索された料理店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．実際に食したであろう人たちの料理の感想は非常に有用であり，店舗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．また飲食店を利用した後，その店の感想を口コミとして書き込むことも一般的になっている．料理についての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．その回答もグルメ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店を決めるユーザは多い．

しかしながら，グルメサイトに登録されている飲食店は非常に多く，かつ，飲食店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．実際，書き込みが多い店では数百件～数千件もの書き込み数になる場合もある．数件程度の書き込みであれば書き込みを一つ一つ見ていくこともできるが，数百件ともなると読むには膨大な時間がかかってしまう．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店やその店の料理の良し悪しを判断せざる

[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

を得ないことが多い．

また口コミには多種多様な情報が含まれており，行きたい店を見つけるのに不要な情報も多い．自分が知りたい内容にたどり着くにはなおさら多くの時間がかかってしまう．こうして自分にとって有用な書き込みがあったとしても，それに気づかないで膨大な書き込みの中に埋もれてしまう可能性もある．

以上の問題に着目し，著者らはグルメサイトに登録された口コミ情報を利用し，効率的かつ効果的に希望する飲食店または口コミ情報を見つけることができるシステムを構築することを目的とした．そして特に本研究においては，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ようにすることを目標に定めた．その店の料理が美味しいかどうかは，特に口コミ情報に頼る場合が多いと思われる．店員に面と向かって言うことが少ない料理の味付けや素材の鮮度に関する感想が口コミ文章の中に含まれていることもしばしばあり，料理の味については店から提供される情報より，口コミ情報の方が信頼できる可能性が高い．

以上の目標に沿い，Web サービス「食探（しょくたん）」を開発した．食探を利用する際，ユーザは通常のグルメサイトに入力するような地域や料理ジャンルなどの基本条件を指定することに加え，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情報を優先的に表示し，かつ，長いレビュー文章をそのまま表示する代わりに，レビュー文章から抽出した料理に関する評価文を短い要約文で作成して表示する．これらの機能により，膨大な口コミから有用な情報を効率良く探し出すことができるようになっている．

2. 関連研究

グルメサイトには大量の飲食店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 [1] は，ゲームレビューサイトを対象として，ゲームのレビュー文章を Word2Vec[17] 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 は隠れ層と出力層の 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．

梅木ら [2] は，料理レシピにおける代替食材の推薦のために Word2Vec と Doc2Vec[15][16] 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，料理のレシピ文章を Doc2Vec を用いて機械学習する．Word2Vec 同様に Doc2Vec も隠れ層と出力層の 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．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するようになっている．

次に，長い投稿レビュー文章を短い文章に要約する関連研究について述べる

中野ら [3] は，通販サイトにおける投稿レビューを要約するために，レビュー文章から抽出した属性と意見のペアからなる短い要約文を作成して表示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および係り先のペアを属性と意見のペアと見なし，それを要約文としてユーザに提示している．

紀本ら [4] も，映画レビューサイトを対象として，レビュー文章をそのまま表示する代わりに，レビュー文章から抽出した属性と意見のペアからなる短い要約文を作成して表示する方法を提案している．

3. 提案

グルメサイトに登録された口コミ情報を利用し，自分の希望する料理店を手早く探すことができる Web サービス「食探」を提案する．ユーザがどのような料理が食べたいかに関する希望や質問を日本語文で入力すると，システムはユーザの入力文に類似した口コミ文章が多い飲食店を検索し，料理に関係する口コミ情報を優先的に選んだ上で要約してユーザに表示する．

本システムは，インターネットに公開されているライブドアグルメのデータセット [6] を利用して構築されている．このデータセットは学術研究用に公開されているものであり，飲食店 21 万 4 千件，口コミ数 20 万 6 千件の情報が入っている．2011.4.20 時点のスナップショットデータであり，すべてをダウンロードして利用することができる．

以下に，本システムの機能の概要を述べる．

(1) 類似口コミ文の検索：データセット内の全口コミ文章を Doc2Vec に入力してベクトル分散表現を予め作成しておく．料理店を検索する際にユーザが入力した入力文のベクトル分散表現と近いベクトルを持つ口コミ文章を検索すると共に，そのような口コミが多い飲食店を抽出する．

(2) 要約文の作成：(1) で検索された口コミ文章を日本語

係り受け解析器 CaboCha[9][10] を用いて短い係り受け文に要約する．たとえば「太郎は花子が読んでいる本を次郎に渡した」という文は「太郎は渡した」「花子が読んでいる」というような係り受け文（修飾関係の解析結果）に変換される．CaboCha は，日本語形態素解析エンジン MeCab[8] の開発者によって開発されたオープンソースソフトウェアであり，日本語文内の語句の係り受け関係を解析する機能を提供している．

(3) 料理情報の抽出：(2) で作成された係り受け文の中から，料理に関係する口コミ情報を抽出してユーザに提示する．具体的には，料理名が含まれている係り受け文，または，料理を評価する用語が含まれている係り受け文を抽出するようにした．料理名は各種公開データベースソース [13][14] から取得した．また料理を評価する用語については，早川ら [12] の，おいしさを評価する用語（官能評価のために用語選定された 445 語）の研究成果を利用した．たとえば料理名としては「ハンバーグ」「唐揚げ」「アイスクリーム」「パンチェッタ」「チリンドロン」というように家庭で食べることが多いものから普通はレストランで食べるものまでを含んでいる．また料理を評価する用語には，ふわふわ，ふんわり，パリッ，パリパリ，サクサク，粒状，乳状，水飴状，などの用語が含まれている．

(4) 評価文の抽出：(3) で抽出された係り受け文の中には，料理名は入っているがその料理に対する感想や評価が書かれていない文が多数混じっている．そこで，日本語評価極性辞書 [7] に登録されている語句（評価語）を含む係り受け文のみを抽出してユーザに提示するようにした．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．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 5278 組が登録されているが，ポジティブ極性を持つ評価語を含む係り受け文に限定してユーザに提示するようにした．たとえば日本語評価極性辞書に登録されているポジティブ評価語には，おいしい，うまい，充分だ，十分だ，十二分だ，上々だ，上級だ，上出来だ，などが含まれている．

本システムは Python[19] 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．MeCab と CaboCha はそれぞれの Python 用インターフェースをサーバにインストールして用いている．

4. 実装

食探の実装について，図 1 を参照しながら述べる．

図中，(1) から (6) までがシステムにデータセットを登録する際の処理である．この処理は，全飲食店の全口コミ文章を Doc2Vec に入力してベクトル分散表現を作成するための (1) から (3) の処理と，個々の口コミ文章を日本語係り受け文を作成し，その読みをデータベースに記録するための (4) から (6) の処理に大きく分かれる．一方，(7) から (9) はユーザが料理店を検索する際の処理である．

(1) (3) では，1 つの飲食店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，すべての飲食店についての文章（飲食店毎にまとめられた口コミ文章）と文書 ID（飲食店 ID に相当）を Doc2Vec に入力する．このため (1) では，飲食店一つ一つについて，書き込まれた口コミを 1 つの文章にまとめる前処理を行う．

(2) 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 (2) で MeCab の分かち書き機能を利用する．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 [11] を利用した．また，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[5] 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．

(3) 全飲食店の全口コミ文章を Doc2Vec に入力してベクトル分散表現を作成する．Doc2Vec は文書のベクトル分散表現を獲得するための手法であり Word2Vec の拡張モデルである．Word2Vec が入力として単語列を受け取るのに対し，文章 ID を併せて入力することで，単語の分散表現だけでなく文章の分散表現の獲得が可能となっている．本実装では Python のライブラリである gensim[18] 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 300 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．こうして飲食店ごとに 300 次元のベクトル分散表現を作成してファイルに保存する．

(4) (5) では，個々の口コミ文章から日本語係り受け文を作成するために CaboCha を利用する．CaboCha に入力する文章は句点から句点までのような単位文である必要があるため，(4) では各口コミ文章を複数の単位文に分割する前処理を行う．単位文は，句点や「！」，「？」記号，および，2 つ以上連続したスペース記号，改行コード等を終点として文章を分割して得られるものである．

(5) (4) で得られた単位文を日本語係り受け解析器 CaboCha を用いて短い係り受け文に変換する．たとえば「太郎は花子が読んでいる本を次郎に渡した」という文を CaboCha に入力すると，「太郎は渡した」「花子が読んでいる」「読んでいる本を」「本を渡した」「次郎に渡した」という係り受け文（修飾関係の解析結果）が出力される．ここで作成された係り受け文を

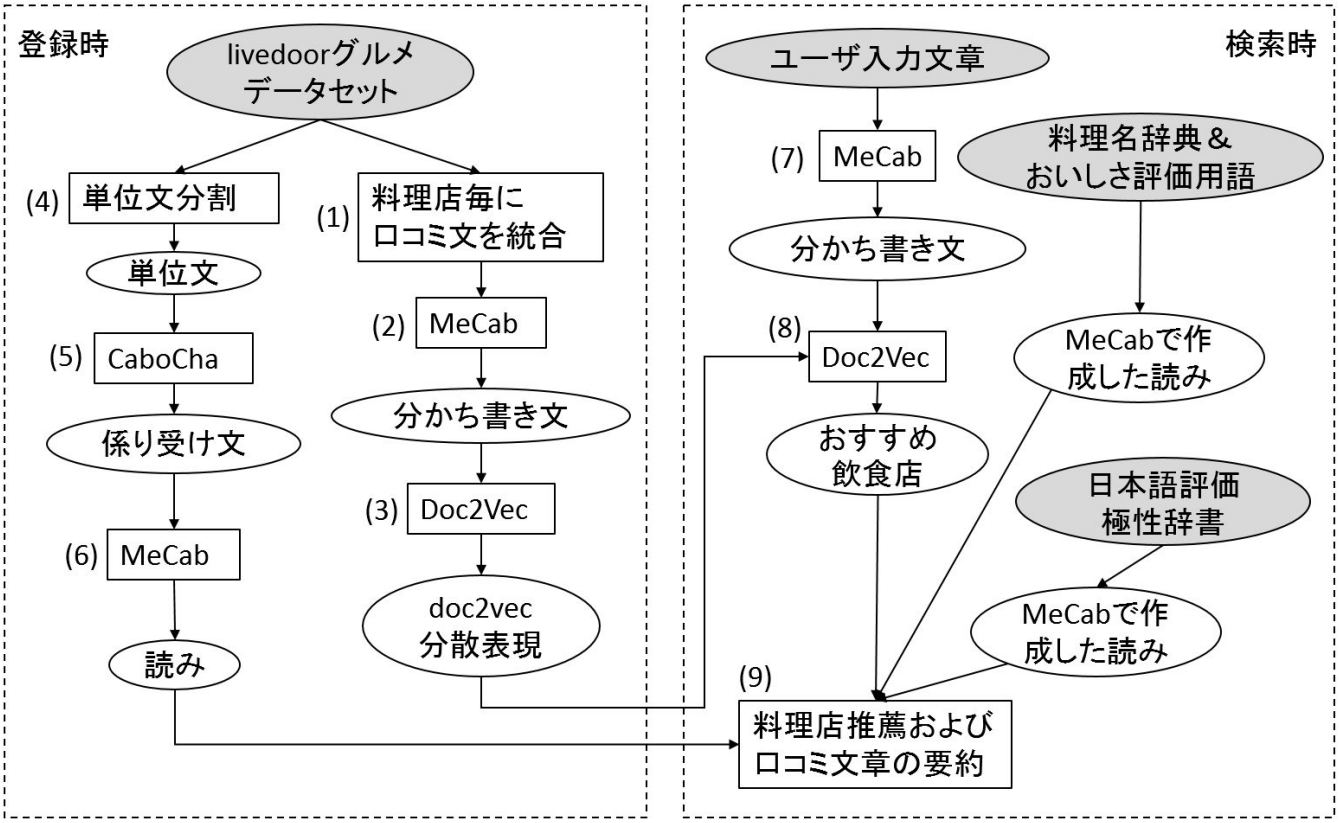

図 1 システム構成

データベースに保存するが，ユーザに提示される要約文の候補となる．

(6) (5) で作成された係り受け文には，料理に関係のないものが多数存在する．料理店検索時にはそれらを除外してユーザに提示する必要があり，(9) では，料理名が含まれている係り受け文，または，料理を評価する用語が含まれている係り受け文を抽出する処理を行うが，(6) ではそのための前処理を実施する．

日本語には表記ゆれが多数存在するためにその対策の必要があり，動詞や形容詞などの活用語については MeCab を用いて原型を取得するようにした．例えば，美味しかった，美味しくなる，美味しければ，はすべて原型の「美味しい」に変換する．加えて，漢字/ひらがな/カタカナ，全角半角の違いを吸収するために，MeCab を利用して語句の読みを取得して全角カタカナで登録するようにした．例えば，「美味しい」「おいしい」は全角カタカナの「オイシイ」に変換されてデータベースに登録される．(2) 同様，MeCab 辞書には mecab-ipadic-NEologd を利用した．

(7) ユーザは，どのような料理が食べたいかに関する希望や質問を日本語単文で入力する．この入力文は (8) で Doc2Vec に入力するため，(7) では MeCab を利用して分かち書き文に変換する．(2) 同様，MeCab 辞書には mecab-ipadic-NEologd を利用した．またストップワード除去のために SlothLib を用いた．

(8) 分かち書きされたユーザ入力文を Doc2Vec に入力してベクトル分散表現を作成し，そしてこのユーザ入力文のベクトルと距離が近いベクトルを持つ文書（飲食店毎にまとめられた口コミ文章）の文書 ID（飲食店 ID に相当）を，(3) で作成したベクトル分散表現のファイルの中から抽出する．ユーザには，ここで抽出した店をおすすめ飲食店として提示する．

(9) (6) で作成した全飲食店の係り受け文の読みの中から，(8) で抽出されたおすすめ飲食店に対応するものだけを抽出し，さらに，料理名が含まれている係り受け文，または，料理を評価する用語が含まれている係り受け文のみを選び出す．

次にこれらの中から，日本語評価極性辞書に登録されている語句（評価語）を含む係り受け文のみを抽出してユーザに提示する．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 5278 組が登録されているが，ポジティブ極性を持つ評価語を含む係り受け文に限定してユーザに提示するようにした．

なお，各種公開データベースソースから取得した料理名，料理を評価する用語，日本語評価極性辞書内の用語については，(6) で作成した全角カタカナ化された係り受け文とマッチングさせるために，それぞれ MeCab を用いて原型を取得してその読みを全角カタカナに変換して保存しておいたものを用いた．これらの処理により，料理に対するポジティブな感想や評価が書かれている係り受け文のみを抽出してユーザに提示することが可能となった．

具体的にシステムの実際の出力例を用いて動作を説明する．(5) および (6) で作成した全飲食店の係り受け文に「美味しい 魚料理でした」「真鯛の カマは」「ブログにて 公開しています」が含まれていた場合を考える．

(9) の処理において，「ブログにて 公開しています」は料理名も料理を評価する用語も含まないため候補から外れ，「美味しい 魚料理でした」「真鯛の カマは」が残る．

つぎに，日本語評価極性辞書内の用語との比較を行うが，「真鯛の カマは」については 「真鯛の」 の部分も 「カマは」 の部分も日本語評価極性辞書内の用語とマッチングしない．店で出される真鯛のカマが美味しいのかどうかはこの係り受け文からはわからないため，美味しい料理が食べられる料理店を手早く探すという観点から除外すべきである．よって，「真鯛の カマは」が候補から外れて「美味しい 魚料理でした」が残り，これがユーザに提示されることとなる．

以上の，料理名または料理を評価する用語との比較も，日本語評価極性辞書内の用語との比較も，実際は，分かち書きされた活用語の原型を全角カタカナ読みに変換して実行するため，「おいしい さかな料理 でし た」や「美味しい 魚料理 です」であっても候補から外れることはない．

本システムを実装して検証したところ，平均的に要約作成機能によって，口コミ文章がほぼ 1/10 の文字数に要約されることがわかった．

5. 評価実験

試作したシステムを Web サーバ上に構築し，大学生 10 名に使用させる実験を行った．データベースには，飲食店 21 万 4 千件，口コミ数 20 万 6 千件の情報が 入っている状態である．

システムの画面を図 2 に示す．ページの最上部に入力欄が設けられており，入力欄（日本語自然文で入力），送信ボタン，出力形式指定（口コミ全文表示／口コミ要約表示），出力件数指定，店番号入力欄（通常は空白）が配置されている．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．

『美味しい料理が食べられる料理店を手早く探すためのシステムです．入力欄には，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してください．例えば，「鹿肉のキノコ添えソテーという肉料理を食べたい」と入力して送信ボタンを押します．すると件数欄で指定された件数のおすすめ店舗が出力されます．出力形式は，口コミ全文表示または口コミ要約表示が選べます．なお，店番号の欄に店番号をいれて送信ボタンを押すと，おすすめ店舗検索ではなく，その店の口コミが表示されます．この場合も出力形式は，口コミ全文表示または口コミ要約表示が選べます．』

5.1 要約機能の検証

レビュー文章から抽出した料理に関する良い評価文を要約して表示する機能が本稿において提案する部分であるため，まず要約機能によって手早く美味しそうな料理を探すことができるようになるのかどうかを評価することとした．被験者全員に特定の店番号を店番号欄に入力させ，全員に同じ口コミ文章を見せて実験を行った．

被験者の半分には，店舗 A について，全文表示，要約表示の順で表示して，表示された文書をすべて読むのにかかる時間を回答させ，つぎに，店舗 B について，要約表示，全文表示の順で表示し，表示された文章をすべて読むのにかかる時間を回答させた．また残りの半分の被験者には，逆に，店舗 A について，要約表示，全文表示の順で表示して，表示された文書をすべて読むのにかかる時間を回答させ，つぎに，店舗 B について，全文表示，要約表示の順で表示し，表示された文章をすべて読むのにかかる時間を回答させた．

店舗 A については全文表示時 4483 文字，要約表示時 358 文字が表示され，店舗 B については全文表示時 5236 文字，要約表示時 438 文字が表示されていた．なお，流し見にならないよう，被験者には「読んだ結果，何を食べたくなったか回答してください」と指示した．

実験結果について述べる．

表示されたレビュー文章をすべて読むのにかかった時間の平均は，全文表示時が 341.6 秒，要約表示時が 54.3 秒であり，要約機能によって時間が 15.9 %に短縮されることがわかった．時間を大幅に短縮できたことから，手早く美味

図 2　システム利用画面

しそうな料理を探すという観点からは一定の効果があったと考えられる．これに関する被験者からのコメントを以下に示す．

- (要約表示時) 料理についての部分が簡略表示されているので興味のある料理は早く見つけることができました
- (要約表示時) 全文よりも内容が頭に入ってきやすく読みやすかった
- (全文表示時) 文字数が多いため本当に食べたい料理かを判断するのに時間がかかりました
- (全文表示時) 長い文章を読み進めて他に食べたくなる料理があるかを探すのは時間がかかりました

一方で，文字数が約 8 %に短縮されていたにもかかわらず，時間は約 16 %の短縮であった．これに関係すると思われる被験者からのコメントを以下に示す．

- (要約表示時) 料理名が欠落している文があり，何の料理についてかわからず悩むことがあった
- (要約表示時) 語尾が切れていたりして通常の日本語ではないため読んでいて違和感があった

作成された要約文は元の全文より読みにくかった可能性があり改良の余地がある．

5.2　実使用検証

次に前述した実験と同じ被験者に食探を実使用させ，システムの使用感を検証した．

被験者全員には，入力欄に自分の好きな内容を入力させた．被験者には「入力欄にどのような料理が食べたいかに関する自分の希望を入れて，送信ボタンをおしてください」と指示をした．今回は，地域を東京，料理ジャンルを西洋料理と設定した以外は自由にシステムを使用させた．そして使用後に要約表示についての感想をインタビューした．

回答を以下に示す．

- 検索した料理は見つからなかったが，ほかの料理についての評価がすぐに読めるのでそちらの料理を食べたくなった
- 自分が探す努力をしなくていいのでいいと思いました．
- 具体的な料理名が出ていない文については料理名も出るとよかった

システムの動作を解析したところ，具体的な料理名が欠落している文がある問題については，その文の中に，抽出した係り受け文とは別の係り受け文に料理名が含まれていることが多かった．具体的な料理名を係り受け文に追加することについては今後の課題としたい．

5.3　個別機能の検証

食探が提供する要約機能は，係り受け文への変換，料理情報の抽出，ポジティブ評価語の抽出という個々の機能を組み合わせた構成となっている．個々の機能によって口コミ文章がどの程度短縮されるかついて検証を行った．前述の実験でも使用した店舗 A と店舗 B の口コミ文章を用いて検証した．以下で説明する文字数は店舗 A と店舗 B の平均である．

前述した通り，全文表示時は 4860 字，食探（係り受け文変換＆料理情報抽出＆ポジ評価語抽出）を使用した場合は 398 字であり，全文表示の 8 %に減る結果となった．

一方，食探システムの個々の機能を ON/OFF した結果は以下の通りとなった．

- 係り受け文変換のみ 8576 字
- 係り受け文変換＆料理情報抽出のみ 2260 字
- 係り受け文変換＆ポジティブ評価語抽出のみ 1864 字
- 係り受け文変換＆料理情報＆ネガポジ評価語抽出 598 字

係り受け文変換のみの場合は，全文表示の 1.8 倍に文字数が増えた．たとえば「太郎は花子が読んでいる本を次郎に渡した」という文から係り受け文を作成すると，「太郎は渡した」「花子が読んでいる」「読んでいる本を」「本を渡し

た」「次郎に渡した」という 5 つの係り受け文が作成されるが，これらの文字数を足し合わせると元の文章より多くなるためである．

係り受け文変換と料理情報抽出のみを行った場合は，文字数は全文表示の約 46 %となった．評価文抽出を行っていないため出力には「真鯛の カマは」「コンポートハーブと ジェラートと」のような係り受け文が含まれた．

逆に，係り受け文変換とポジティブ評価語抽出のみを行った場合は，全文表示の約 38 %となった．「外観で，感じられて」「期待で ワクワクですよ」「組み合わせが 良く」といった，料理に関係があるのか無いのか不明な係り受け文が多数含まれていた．

最後に，食探の評価文抽出で日本語評価極性辞書に登録されているポジティブ評価語とネガティブ評価文の両方を抽出するようにした．この結果 598 字となり，全文表示の約 12 %となった．出力例を読んだ感想としては，ネガティブ評価文であっても受け取り方によってはそれ程印象が悪くないものもあった．たとえば，「お米の 残り具合なんて」,「その季節は 痩せ気味なので」などが，ネガティブ評価文にマッチングした係り受け文である．ネガティブ評価文を部分的に食探に組み込むことも今後検討したい．

6. まとめ

本稿では，美味しい料理が食べられる料理店を手早く探すためのシステム 食探 を提案した．レビュー文章から抽出した料理に関する良い評価文を要約して表示する機能が特徴である．

要約機能によって手早く美味しそうな料理を探すことができるようになるのかどうかを実験により評価した結果，表示された文章をすべて読むのにかかった時間の平均は，全文表示時が 341.6 秒，要約表示時が 54.3 秒であり，要約機能によって時間が 15.9 %に短縮されることがわかった．時間を大幅に短縮できたことから一定の効果があったと考えられる．

しかし一方で，「具体的な料理名が欠落している文があり，何の料理についてかわからず悩むことがあった」というコメントもあり，今後これらの問題に対応したい．

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] 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，,
[2] 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)
[3] 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)
[4] 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)
[5] 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)
[6] livedoor グルメの DataSet，学術研究用アーカイブ，2011.4.20 時点 http://blog.livedoor.jp/techblog/archives/65836960.html (2019)
[7] 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)
[8] 工藤：MeCab - Yet Another Part-of-Speech and Morphological Analyzer, http://taku910.github.io/MeCab/ (2019)
[9] 工 藤：CaboCha - Yet Another Japanese Dependency Structure Analyzer, http://taku910.github.io/CaboCha/ (2019)
[10] 工藤，松本：チャンキングの段階適用による日本語係り受け解析，情報処理学会論文誌，Vol.43，No.6，pp.1834-1842 (2002)
[11] mecab-ipadic-NEologd : Neologism dictionary for MeCab, https://github.com/neologd/mecab-ipadic-NEologd/blob/master/README.ja.md (2019)
[12] 早川: おいしさを評価する用語，官能評価のために用語選定された 445 語，日本調理科学会誌 Vol.41, No.2, pp.148〜153 (2008)
[13] ルーラル電子図書館 日本の食生活全集 データベース http://lib.ruralnet.or.jp/yougo/dentou/a.phtml (2019)
[14] 料理名一覧：建帛社 https://www.kenpakusha.co.jp/pdf/pdf_ee7.0_ryouri_ichiran.pdf (2019)
[15] 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)
[16] models.doc2vec - Doc2vec paragraph embeddings, https://radimrehurek.com/gensim/models/doc2vec.html (2019)
[17] models.word2vec - Word2vec embeddings, https://radimrehurek.com/gensim/models/word2vec.html (2019)
[18] gensim, Free Python Library, https://radimrehurek.com/gensim/ (2019)
[19] https://www.python.org/ (2019)