Word2Vec と Web 検索を用いた検索クエリ置換手法

鹿島　好央†　　北山　大輔†

† 工学院大学 情報学部　〒 163-8677 東京都新宿区西新宿 1-24-2
E-mail: †j113029@ns.kogakuin.ac.jp, kitayama@cc.kogakuin.ac.jp

あらまし　Web 検索のクエリ入力において，対象オブジェクトの具体的な名称を知らないために検索を行えないという状況や，曖昧なクエリを入力してしまい目的の情報が得にくいという問題が起こりやすい．これらの検索クエリの精度向上のためのクエリ置換の研究はこれまでも行われてきたが，クエリログなどの大規模なログデータを必要とする研究が多かった．本論文ではクエリ置換に Word2Vec を用いることに着目する．このことによって大規模なログデータがなくても置換が行えるようになることを期待する．また Word2Vec によるクエリ置換を行う上で問題となりうる非学習単語への対応を行うことで様々な検索クエリに対応できるクエリ置換方法を提案する．

キーワード　Word2Vec，Web 検索，クエリ代替

1. はじめに

近年，Web 検索の需要が非常に高まっている．検索を行う上でより良い情報を得るためには適切なクエリを入力する必要があるが，具体的な名前を知らない場合や冗長なクエリを入力している場合など，適切なクエリを入力していない，入力が難しい場面というのは存在する．例えばある日突然見かけた鳥について検索したいと思った場合，その鳥の名前を知らなければユーザは検索を行うことができない．検索するとしても “鳥 and 茶色” のように具体的ではないクエリになってしまうと考えられる．

実際に “鳥 and 茶色” という検索クエリで検索を行うと鳥の一覧が見られる Web サイトをいくつも見つけられる．これらのサイトに存在する情報を活用することで十分に鳥の名前を把握することは可能である．しかしユーザが鳥の名前を知りたいのではなく，鳥の名前を用いた検索を行いたい場合，“鳥 and 茶色” で検索して検索結果から名前が載っていると思われるサイトにアクセスを行い，サイト内で自分の求めている鳥の名前を見つけるという一連の工程は非常に時間がかかる．

検索クエリに対する研究としては検索クエリに単語を追加することで，より素早く目的の情報に到達できる検索クエリ拡張が存在する．検索クエリ拡張を行うことで検索結果から目的のサイトを見つけやすくなる．例えば “鳥 and 茶色 and 小さい” といった検索クエリにすることでユーザが求めている鳥の名前を見つけるのも容易になると思われる．しかし検索クエリ拡張では元のクエリよりも精度の高い検索結果が得られるが，検索する一連の流れが残ってしまう．

このとき，“鳥 and 茶色”(注1)の検索を行う時点でクエリに対応する単語を知ることができれば，ユーザは検索して必要な名称を得る手間を減らすことができる．また 1 単語に絞れない場合でも，複数単語を一覧として提示することでユーザはサイトを探す必要がなくなり，適切な情報が書かれているページを探す手間が省ける．また提示する際に画像を同時に出すことで，慣れたデザインで情報を探せるようにすることも可能である．そのため，具体的な名称が分からない場合に，入力したあいまいなクエリの代替候補を推薦し，置き換えることで精度向上が図れると考える．

一般的にクエリの代替候補の取得には単語ベクトルを用いた類語取得を行える Word2Vec を利用することが考えられる．Word2Vec は，Mikolov ら [1] によって提唱された，ニューラルネットワークを用いて単語の分散意味表現を計算する手法，およびそのオープンソース実装の名称である．Word2Vec では「同じ文脈で出現する単語は類似した意味を持つ」というハリスの分布仮説 [2] に基づき，「ある単語列が与えられたとき，次に出現する単語を予測する」というタスクをニューラルネットワークに学習させることで，文脈や単語の語順を考慮した単語の分散表現を得ることができる．しかしながら，Word2Vec では非学習単語の類語や複数単語からなるクエリの類語を取得することができない．そのため本論文では Word2Vec と Web 検索を組み合わせた類語取得手法について提案する．

以降，2 章では関連研究と本研究との差異について述べ，3 章では Web 検索と Word2Vec を用いた類似語推薦手法について述べる．4 章では実験および考察について説明する．そして，5 章でまとめと今後の課題について述べる．

2. 関連研究

入力された検索クエリに対する推薦を行う研究としては検索クエリ拡張と検索クエリ置換が存在する．検索クエリ拡張を行う研究としては He ら [3] のようにユーザのクエリログやクリックログを基に推薦を行うものや，Kraft ら [4] のように意味ネットワークを用いて検索意図を推定し，検索クエリを拡張する手法などが存在する．しかし 1 章でも触れたが，検索クエリ拡張では検索処理そのものを減らすことはできない．本稿では検索によって得られる情報を基に再度検索することを想定するため，検索処理を減らすことができる検索クエリ置換を採用する．

検索クエリの置き換えについては今井ら [5] や Mei ら [6] の

(注1)：以後，and 検索は単なるスペース区切りで表す．

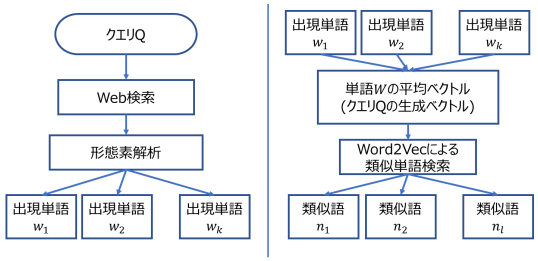

図 1　検索クエリに対する Word2Vec を用いた類似語取得の流れ

ように検索結果のクリックログを基に置き換えを行うものや，Chein ら [7] のように検索クエリログを基に置き換えを行うものは多く存在する．しかしこれらには膨大なログデータが必要であり，ローカル環境のモデルデータのみで置き換えを行う本稿とは異なる手法である．Wang ら [8] は，確率的翻訳モデルと確率的文脈モデルを用いて，準同義語や共起しやすい単語を分析し，クエリを改良する手法を提案した．共起関係を分析して行う点は Word2Vec と類似しているが，本稿では Word2Vec を用いることでより容易に実現できるかを調査する．また Word2Vec を検索クエリに適用させる試みを Fernando ら [9] が行っている．しかし英語の学習データを用いた研究であり，単語の区切りの少ない日本語への性能は未知数であるため，日本語での学習結果を基にクエリ置換を行う本稿とは学習データと適用する言語に違いがある．

3. 提 案 手 法

本論文で提案する Web 検索結果を用いた Word2Vec による類似語推薦の流れを説明していく．

3.1 Word2Vec と Web 検索を用いた置換クエリの抽出

Word2Vec では，各単語にベクトル表現を与えることで，ベクトルの値が類似する単語を得ることができる．しかし，Word2Vec では検索クエリのように複数単語の組み合わせに対するベクトルを持つことができない．つまり “鳥”や “茶色”という単語に対するベクトルは存在するが，“鳥 茶色”というクエリそのものを表すベクトルは Word2Vec では得ることができない．ベクトル表現が得られない以上 Word2Vec を用いて類似語を得ることもできない．そのため，“鳥 茶色”に対する仮のベクトルを生成することによって，Word2Vec を用いて類似語を取得できるようにする．このとき生成するベクトルは，クエリに関連する単語のベクトルの平均を取ることで表現する．この方法で生成すれば少なくともベクトル生成に使用したクエリに関連する単語と関係性のあるベクトルが生成できる．また生成ベクトルをうまく生成できた場合，検索結果には表れていない類似語を得ることができるようになる．

クエリに対する仮のベクトルを生成するため，クエリに関連する単語が必要となる．この単語取得に Web 検索を用いることで，学習されていない複数単語や，新語などに関連する単語を得ることができる．

クエリに対するベクトルを生成し，類似語検索を行う流れは次のようになる．

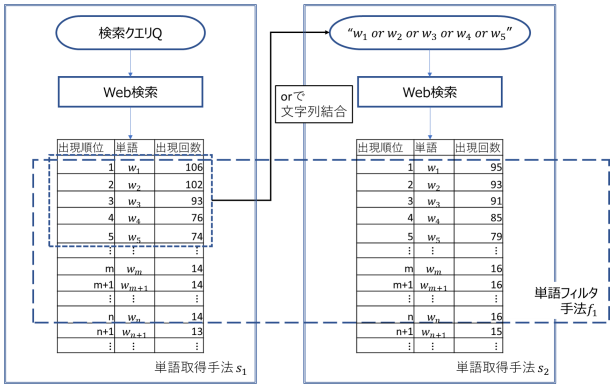

図 2　各取得手法と f_1 における単語取得の流れ

(1)　検索クエリを基の Web 検索を行い，検索結果に含まれる単語を抽出する

(2)　得られた単語について，各単語の Word2Vec におけるベクトル表現を取得し，単語の平均ベクトルを生成する

(3)　得られた平均ベクトルを検索クエリのベクトルとして，Word2Vec で類似語検索を行い，得られた類似語をクエリの類似語として提示する

この流れを表したものが図 1 となる．3.2 章において (1) となる Web 検索を用いた関連語抽出について，3.3 章において (2) となる関連語集合を用いたベクトルの生成についての詳細を解説する．

3.2 ベクトル生成のための関連語抽出

ベクトルを生成する上で，Web 検索を用いて関連語を取得する必要がある．単語の取得手法について，1 つのクエリに対して次の 2 つの取得手法を提案する．

s_1)　単語 Q をクエリとして Web 検索を行い，検索結果のスニペットとタイトルから単語を取得する

s_2)　手法 s_1 で取得した単語の中から，出現回数が最も高いものを 5 つ選択して or で結合したものをクエリとして Web 検索を行い，検索結果のスニペットとタイトルから単語を取得する

また，s_1, s_2 にて得られた単語集合を次の手法でフィルタすることで，より精度の高いベクトルを生成することを狙う．

f_1)　単語の出現頻度が上位 n 位である単語のみを使用する

f_2)　単語集合の重心との類似度を基に，外れ値検定を行い外れ値を除去する

f_3)　外れ値を除去した f_2 の結果から，さらに重心との類似度を基に，重心に近い上位 n 件の単語を使用する．

出現頻度の少ない単語については，元の単語とあまり関係がない場合が多い．このため，出現頻度の低い単語を削除することでノイズとなる単語を削除することを狙う．また，ベクトル分布に基づいて外れ値を探索することで，重心を生成する上でノイズとなりうる単語を削除できる．

図 2 では取得手法 s_1, s_2 による単語取得の流れと，フィルタ手法 f_1 で選択する単語について示している．

フィルタ手法 f_1 では，検索結果に出現した回数を基に，出現回数が多かった上位 n 位までの要素を使用する．出現回数が同率である場合，単語間に優位性はないと考えられるため，次

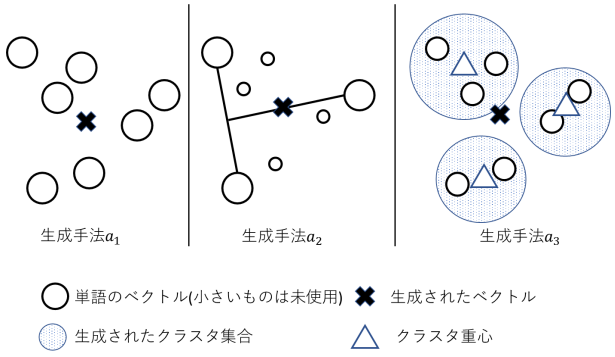

図 3　a_1〜a_3 における各ベクトル生成手法の概念図

の条件のもとに単語を選択する．

- 単語出現回数が上位 n 位以内に含まれる
- 単語出現回数が n 位の要素と同回数であれば，m 個以上となっても単語生成に使用する
- 使用する単語集合が n 個以下である場合は出現回数を問わずすべて使用する

3.3　関連語を用いたクエリベクトル生成

3.2 章において得られた関連語を基に，類似語検索を行うためのベクトルを生成するベクトル生成手法としては次の 3 つを考える．各生成手法の概念図が図 3 となる．

a_1)　関連する単語のベクトルの平均ベクトルを生成する．

a_2)　関連する単語のベクトル集合から，最も距離が遠くなるように 3 点を取得して平均ベクトルを生成する．

a_3)　K-means クラスタリングを行い，各クラスタの重心を取得して平均ベクトルを生成する．

生成手法に関して，a_1 については Word2Vec そのものの手法としても使われており，ユーザの行動履歴を Word2Vec に学習させるという試み (注2)においても有効であるとされている．単語集合に外れ値が含まれていても大部分が正しい値であれば精度の高いベクトルを作れるが，多くの単語が偏って分布している場合，単語集合の偏りに大きく影響を受けたベクトルを生成してしまう欠点がある．a_2 については，距離の遠い 3 点を選択することで，単語分布の偏りを回避して作成することができる．しかし選択する単語一つ一つの影響が大きく，外れ値を使って生成してしまうと精度が下がりやすいという問題もある．a_2 の問題に対し，a_3 ではクラスタリングを行うことで外れ値の影響を抑えることができる．手法 a_1, a_2 の 2 手法については Word2Vec の類似語集合を用いた生成においてどちらも同程度に精度が高いベクトルが生成できることが 3.4 章に記す予備実験により判明している．

また，3.2 章において定義したフィルタ手法と生成手法の組み合わせの関係性は大きいと考えられるため，ベクトル生成手法は先ほどの a_1〜a_3 とフィルタ手法 f_1〜f_3 を組み合わせた以下の a_1〜a_3· f_2 として定義する．このとき Web 検索結果から得られる単語集合を W とする．

a_1)　W のすべての単語を用いた平均ベクトル

(注2)：http://www.slideshare.net/recruitcojp/ss-56150629

表 1　Word2Vec の類語を基に生成したベクトルの元の単語との類似度

	手法 a_1	手法 a_2
最大値	0.948	0.946
最小値	0.439	0.455
平均値	0.752	0.748
中央値	0.759	0.752

a_1· f_1)　W から出現頻度の少ない単語を除いた単語の平均ベクトル

a_1· f_2)　W の重心との類似度を基に外れ値検定を行い，外れ値を除いた単語集合の平均ベクトル

a_1· f_3)　W の重心との類似度を基にした上位 n 件の単語を用いた平均ベクトル

a_2· f_2)　外れ値検定を行い，最も距離が遠くなるように 3 点を取得して生成した平均ベクトル

a_2· f_2f_1)　外れ値検定を行い，出現頻度が少ない単語を除いた単語集合から，最も距離が遠くなるように 3 点を取得して生成した平均ベクトル

a_3)　W に対して k-means クラスタリングを適用し，得られた各クラスタの重心の平均ベクトル

a_3· f_2)　W の重心との類似度を基に外れ値検定を行い，外れ値を除いた単語集合に対して k-means クラスタリングを適用し，得られた各クラスタの重心の平均ベクトル

a_1· f_1〜a_1· f_3 は生成手法 a_1 に f_1〜f_3 をそれぞれ組み合わせたものである．a_2· f_2, a_2· f_2f_1 は生成手法 a_2 にフィルタ手法 f_2 を行ったうえで，f_1 とフィルタなしを組み合わせたものである．フィルタ手法 f_2 を常に適用する理由として，生成手法 a_2 では極端な単語を選択する傾向にあり，外れ値を選択することが多かったため，外れ値を除外する f_2 を常に適用している．a_3, a_3· f_2 は生成手法 a_3 に f_2 とフィルタなしを組み合わせたものである．f_1, f_3 を生成手法 a_3 に適用しない理由として，K-means クラスタリングを行うことから，使用する単語数を少なくすると無理なクラスタを生成しやすくなる．そのためフィルタ手法 f_1, f_3 は除外した．a_1· f_3 は生成手法 a_1 に f_2 を行った上で，単語集合の重心に近い n 個の単語を選択し，ベクトル生成を行う手法となっている．これは実験の際に偶然見つかった手法であり，精度の良い結果を残していたため採用したものである．

3.4　予 備 実 験

予備実験では Word2Vec で学習を行った単語のうち，450 個をランダムに抽出して，各単語の類似度が高い語の集合を基に手法 a_1, a_2 でベクトル生成を行った．このとき生成したベクトルと元のベクトルとの類似度を算出する．生成手法が適切であれば，Word2Vec におけるある単語のベクトルに類似する単語集合を与えることで，元の単語のベクトルに似たものが生成されるはずである．似たベクトルが生成できれば，Web 検索においても関連語を得ることで，クエリに対応するといえるベクトルが生成できる可能性が高い．予備実験の結果を次の表 1 に示す．

表 1 より，ベクトルの生成手法 a_1, a_2 において，単語ベクト

ルの分布による差はあるが，どちらの精度もほぼ同等の平均値が出ていることが分かる．このため，検索結果の単語の分布で適切な生成手法を選択することが可能である．

4. 実　　験

3 章で提案した手法が非学習単語のベクトル生成にどの程度有効であるかを評価実験によって検証する．ベクトル生成手法の評価実験では予備実験で求めた手法と同様の手法で，各手法によるベクトル生成の精度を調査する．4.2 章では提案手法で生成するベクトルと Word2Vec の学習によって得られるベクトルがどれほど類似しているかを調査する．また 4.3 章では，Word2Vec で学習を行った単語に対して提案手法で生成したベクトルによって得られる類似語を取得し，得られた語が類似語として適切なものが取得できているかを調査する．4.4 章では，4.2，4.3 章の結果から生成するベクトルの精度の高い手法を選択したうえで，Word2Vec で学習が行えない複数単語，自然言語を用いて提案手法でベクトルを生成し，得られる類似語が置換候補としてどれほど適切であるかを別の類似語取得手法との差を基に評価する．

4.1 実験の設定

実験にあたり，Word2Vec のモデル作成に日本語 Wikipedia のテキストデータを用いて学習を行った．データの単語単位への分割を行うための分かち書きにはオープンソースの日本語形態素解析器である MeCab [10] を使用した．また Web ページには，様々な新語や流行語が含まれている．MeCab の標準辞書に存在しないこれらの単語に適切に形態素解析を適用するため，Web 上の言語資源から得た新語で標準辞書を拡張した mecab-ipadic-neologd(注3)を単語辞書として用いる．学習を行った結果，1,665,668 単語を含むモデルデータを利用する．クエリによる Web ページの取得には Bing Search API(注4)を用いた．

また，各手法については次の条件を基に実装を行う．

- 生成に使用する単語には名詞のみを採用する
- 同じ単語で実験を行い，同じ検索結果が得られた場合，常に同じ値が得られる実装とする
- ランダムな値を使用する場合は単語のハッシュ値をランダム関数のシード値とする
- 単語生成時，上位 n 件を選択する場合は上位 30 件を使用する (f_1, $a_1 \cdot f_3$)
- K-means クラスタリングを行う際，クラスタ数を 3 とする (a_3)

フィルタ手法 f_2 には次の条件も設定する．

- 外れ値検定にはスミルノフ・グラブス検定を用いる
- 有意水準 10%で検定を行う

4.2 Word2Vec によるベクトルとの比較

ベクトルを生成するにあたって，Word2Vec の学習済み単語のベクトルとどれほど類似したベクトルが生成できるかを評価する．そのため，Word2Vec の単語に対するベクトルを正解データとし，正解データとどれほど類似したベクトルが作成できていたかで評価を行った．各ベクトル生成手法の評価にあたり，Word2Vec のモデル内に存在する単語からランダムに 989 単語を選択し，元のベクトルと生成したベクトルにおける類似度を取得して評価を行う．表 2 に単語取得手法 s_1 における生成手法 a_1〜$a_3 \cdot f_2$ の各生成手法の結果を，表 3 に単語取得手法 s_2 における各生成手法の結果を示す．

表 2　取得手法 s_1 における各生成手法の元のベクトルとの類似度

	a_1	$a_1 \cdot f_1$	$a_1 \cdot f_2$	$a_1 \cdot f_3$	$a_2 \cdot f_2$	$a_2 \cdot f_2 f_1$	a_3	$a_3 \cdot f_2$
最大値	0.760	0.824	0.754	0.917	0.564	0.799	0.625	0.621
最小値	-0.201	-0.248	-0.250	-0.236	-0.164	-0.289	-0.165	-0.083
平均値	0.285	0.295	0.258	0.313	0.176	0.252	0.237	0.301
中央値	0.285	0.288	0.253	0.292	0.173	0.242	0.241	0.309

表 3　取得手法 s_2 における各生成手法の元のベクトルとの類似度

	a_1	$a_1 \cdot f_1$	$a_1 \cdot f_2$	$a_1 \cdot f_3$	$a_2 \cdot f_2$	$a_2 \cdot f_2 f_1$	a_3	$a_3 \cdot f_2$
最大値	0.819	0.856	0.819	0.917	0.653	0.894	0.797	0.797
最小値	-0.180	-0.173	-0.180	-0.229	-0.115	-0.228	-0.146	-0.143
平均値	0.329	0.312	0.329	0.332	0.210	0.298	0.289	0.288
中央値	0.321	0.305	0.321	0.312	0.166	0.276	0.293	0.289

表 4　“トトリのアトリエ”における取得手法 s_1 と s_2 の類似語

	取得手法 s_1	類似度	取得手法 s_2	類似度
検索クエリ	トトリのアトリエ		アーランド or 2 or 錬金術 or 士 or Plus	
類似語	Yahoo!モバゲー	0.592	メルルのアトリエ	0.680
	イースシリーズ	0.586	トトリのアトリエ	0.663
	ゲーム	0.583	エリーのアトリエ	0.643
	コロプラ	0.577	アーランド	0.642
	アトリエシリーズ	0.570	マリーのアトリエ	0.627

元ベクトルとの類似度のみで判断する場合，生成手法 $a_1 \cdot f_3$ が最も精度が高いといえる．また，取得手法については s_2 が s_1 よりも類似度が高い結果となっている．精度が良くなった原因として，単語そのものを使用すると単語そのものに関連する語が得やすいが，単語そのものに類似する語についての情報が得にくい性質があるのに対し，出現頻度の高い単語を用いると単語そのものの情報と，さらに類似する単語に関する情報を得られるようになるためこのような結果になったと考えられる．表 4 では取得手法 s_1 ではあまりよい類似語が出ず，s_2 において類似する語が得られている，ゲーム “トトリのアトリエ”における生成手法 a_1 の類似語とその類似度を示す．

表 4 の類似語から，取得手法 s_1 では “ゲーム”という単語に対する類似語を出しているように見える．取得手法 s_2 では類似する同じシリーズの作品が出てくるようになっている．このような結果となった要因としては “トトリのアトリエ”という検索クエリであると “トトリのアトリエ”に関連する Web ページがヒットし，一つの単語に対する関連単語を収集しすぎていることが挙げられる．“トトリのアトリエ”に対する検索結果から得られた単語は 899 種である．s_2 では単語数が 727 種まで減っている．s_1 では単語数が多すぎたため，結果として特徴が平均化され，ゲームという大きなカテゴリまでしか特定でき

(注3)：https://github.com/neologd/mecab-ipadic-neologd
(注4)：https://azure.microsoft.com/ja-jp/pricing/details/cognitive-services/search-api/

表 5 “東京タワー”における Word2Vec のベクトルの類似語と生成手法 a_1 で得られるベクトルの類似語

Word2Vec	類似度	生成手法 a_1	類似度
[[東京タワー]]	0.413	横浜ランドマークタワー	0.706
電波塔	0.379	サンシャイン 60	0.653
長島ダム	0.374	あべのハルカス	0.651
東京スカイツリー	0.364	[[東京タワー]]	0.637
富士山レーダー	0.352	アトリウム	0.636
イルミネーション	0.344	グランフロント大阪	0.633
大時計	0.344	東京ミッドタウン	0.632
テレビ塔	0.340	オフィスビル	0.626
防潮堤	0.340	東京スカイツリー	0.625
タワー	0.338	クイーンズスクエア横浜	0.621

なくなっていると考えられる．しかし s_2 では，類似する同シリーズの別作品が出ることによって単語数の増加が抑えられている．例えばゲームの攻略情報サイトについて考えても，s_1 では 1 作品についての攻略サイトしか出ないため，攻略情報以外のサイトがヒットするようになる．しかし s_2 では，類似する同シリーズの作品に関する攻略情報サイトもヒットする．そのため単語数の増加を抑えられていると考えられる．実際に s_2 にて得られたサイトについては同シリーズ他作品の攻略情報サイトがヒットしていた．逆に s_1 では “トトリのアトリエ”に対するファン作品と思われるサイトがヒットしている．例えばイラスト投稿サービスである “pixiv”や動画投稿サービスである “ニコニコ動画”が挙げられる．

しかし，出現回数が少なかったなどの要因によって Word2Vec の学習過程で適切なベクトルが生成されなかった場合，提案手法が結果として妥当とみなせるベクトルを生成していても類似度は低下してしまった．表 5 に単語 “東京タワー”における Word2Vec によって得られたベクトルの類似語と生成手法 a_1 によって得られたベクトルの類似語の上位 10 件を示す．このとき，“東京タワー”と “[[東京タワー]]”(注5) は別の単語として扱われている．

表 5 の Word2Vec の類似語について，ある程度の類似性は担保されているが，長島ダムが東京スカイツリーより類似する点や，東京スカイツリーとの類似度が 0.364 と低い数値になっている点などから，“東京タワー”に割り当てられている元ベクトルが妥当ではないといえる．提案手法によって得られたベクトルは Word2Vec のベクトルより妥当であるといえる類似語と類似度を出している．このとき，生成手法で得られたベクトルと Word2Vec の元のベクトルとの類似度は 0.277 と低くなっている．

4.3 学習済み単語に対する生成ベクトルの類似語の評価

生成したベクトルによって得られる類似語について評価を行う．筆者が Word2Vec で学習済みの単語の中から選択した 51 単語について，各手法で得られるベクトルの類似語がどの程度妥当な結果を示しているかを調査した．

生成したベクトルから得られる置換候補語の評価にあたり，

(注5)：Wikipedia の文法で “東京タワー” のページへのリンクを示す記法．

表 6 取得手法 s_1 における各生成手法の類似語の分類

生成手法	A	B	C	D	E
a_1	49.0%	3.9%	2.0%	33.3%	11.8%
$a_1 \cdot f_1$	51.0%	3.9%	3.9%	29.4%	11.8%
$a_1 \cdot f_2$	47.1%	2.0%	2.0%	37.3%	11.8%
$a_1 \cdot f_3$	45.1%	2.0%	2.0%	43.1%	7.8%
$a_2 \cdot f_2$	9.8%	0%	3.9%	31.4%	54.9%
$a_2 \cdot f_2 f_1$	23.5%	0%	2.0%	64.7%	9.8%
a_3	19.6%	0%	0%	33.3%	47.1%
$a_3 \cdot f_2$	17.6%	0%	0%	33.3%	49.0%

各生成手法で得られるベクトルの類似語上位 20 件を使用する．得られる類似語集合について，次の A〜E に分類する．

A) 得られた類似語のうち，5 単語以上が関連性があると判断できる

B) 得られた類似語のうち，5 単語以上が関連性があると判断でき，なおかつ Word2Vec のベクトルが不適当なベクトルになっている

C) 得られた類似語のうち，5 単語以上が関連性があるが，Word2Vec のベクトルとは違う意味を持つ

D) 得られた類似語集合が何らかの意味を持った集合であるが，元の単語との関連性が低い

E) 得られた類似語集合が何らかの意味を持っておらず，単語間の関連性も低い

選択した単語の類似語として妥当な単語が含まれている場合 A，B，C のいずれかになる．A は Word2Vec のベクトルが理想的な結果であり，生成ベクトルがそれに追従する形となっている．B は “東京タワー”の例と同様に，Word2Vec のベクトルが不適当であるが，生成ベクトルが適切な結果となることを示す．C については，多義語において Word2Vec のベクトルと生成ベクトルにおいて違う意味を持ったベクトルになっているが，どちらも類似語として適切であることを示す．D は生成ベクトルが元の単語と違う単語の意味を持ったベクトルを生成したことを示す．例えば “神奈川県”という単語から “観光”の意味を持ったベクトルを生成した場合が D に分類される．E は生成ベクトルの類似語集合が特定の意味を持った集合でないことを示す．E に分類された場合ベクトル生成そのものが失敗している可能性が高い．

実際に 51 単語を A〜E に分類した結果を表 6 に示す．分類にあたっては筆者の判断を基に分類した．関係性の判断が難しい場合，特に類似語集合に元の単語を含んでいたり，きわめて類似している単語を含んでいるが，類似語集合として見ると元の単語の類似語として不適切な場合などは D に分類した．表 6 が Web 検索のクエリに元の単語を使用した結果を，表 7 が元の単語の検索結果の中で最も出現頻度の高い単語 5 個を “or”で文字列結合したものを，Web 検索のクエリとして使用した結果を示す．

表 6，7 ともに生成手法 a_1〜$a_1 \cdot f_3$ の精度が高くなっている．また類似度では精度がよかった $a_1 \cdot f_3$ が表 6，7 ともに a_1 に劣る形となった．これは単語分布の重心に近い単語が一定方向

表 7　取得手法 s_2 における各生成手法の類似語の分類

生成手法	A	B	C	D	E
a_1	62.7%	3.9%	2.0%	27.5%	3.9%
$a_1 \cdot f_1$	52.9%	3.9%	3.9%	33.3%	5.9%
$a_1 \cdot f_2$	62.7%	3.9%	2.0%	27.5%	3.9%
$a_1 \cdot f_3$	51.0%	3.9%	3.9%	37.3%	3.9%
$a_2 \cdot f_2$	25.5%	0%	0%	29.4%	45.1%
$a_2 \cdot f_2 f_1$	43.1%	0%	2.0%	47.1%	7.8%
a_3	21.6%	2.0%	0%	33.3%	43.1%
$a_3 \cdot f_2$	17.6%	2.0%	0%	33.3%	47.1%

表 8　“セッコク”における生成手法 a_1 と $a_1 \cdot f_3$ の類似語

生成手法 a_1	類似度	生成手法 $a_1 \cdot f_3$	類似度
食用菊	0.679	スイセン	0.834
キハダ	0.676	コチョウラン	0.823
シロツメクサ	0.664	ニリンソウ	0.818
ヤマブドウ	0.663	ヤマブドウ	0.810
タラノキ	0.657	ヤマザクラ	0.809
ブナシメジ	0.656	多肉植物	0.803
インゲンマメ	0.654	フクジュソウ	0.799
ギョウジャニンニク	0.653	レンギョウ	0.798
ヨモギ	0.652	ヤブツバキ	0.796
セッコク	0.548	セッコク	0.750

に偏っており，生成ベクトルが一定の方向に引っ張られたことが要因であると考えられる．一定方向に強い意味を持つため元の単語の意味としては違うが，集合としては意味を持つ D に分類されやすくなったことで精度が下がっている．ただし，$a_1 \cdot f_3$ において生成されるベクトルが正しく生成された場合，全ての生成手法の中で最もよい結果になっていた．このために Word2Vec で学習した同一単語のベクトルとの類似度を算出した表 2，3 において最も精度が高くなっていたと考えられる．表 8 では生成手法 $a_1 \cdot f_3$ が適切にベクトルを生成できた，ラン科の植物 “セッコク”における生成手法 a_1 と $a_1 \cdot f_3$ の類似語とその類似度を示す．このときの取得手法は s_2 である．

どちらも類似語に植物を出しているが，生成手法 $a_1 \cdot f_3$ のほうが各類似語との類似度が高くなっている．また a_1 が植物の中でも食べられるものが多いが，$a_1 \cdot f_3$ では花を鑑賞する植物が多くなっている．表 8 のように，生成がうまくいく場合は非常に高い類似度を出すため表 2，表 3 において $a_1 \cdot f_3$ の精度が高くなっていると考えられる．

また，$a_2 \cdot f_2$〜$a_3 \cdot f_2$ では，頻度フィルタを行った $a_2 \cdot f_2 f_1$ 以外ではあまり精度が出ない結果となった．これについては，外れ値対策が適切に行えなかったことによりベクトル生成そのものの失敗が要因として考えられる．実際に $a_2 \cdot f_2$ では生成結果が意味を持たない E に分類されたものが多くなっている．逆に $a_2 \cdot f_2 f_1$ では単語集合が小さいものとなっているため，何らかの意味を持ちやすくなっている．

表 9 に “セッコク” を用いた $a_1 \cdot f_1$, $a_2 \cdot f_2$, $a_2 \cdot f_2 f_1$ における類似語を示す．また表 10 では各生成手法において使用された単語を示す．

表 9　“セッコク”における生成手法 $a_1 \cdot f_1$, $a_2 \cdot f_2$, $a_2 \cdot f_2 f_1$ の類似語

生成手法 $a_1 \cdot f_1$	生成手法 $a_2 \cdot f_2$	生成手法 $a_2 \cdot f_2 f_1$
栽培品種	(株)(注5)	方
サトイモ	方	Dendrobium
ヤマブドウ	(株)	日本
山野草	the	Senecio
園芸品種	株式会社	japonicum
キハダ	(株)(注6)	Asplenium
多肉植物	roles	sieboldii
マメ科	Uncertainty	Asparagus

表 10　“セッコク”における生成手法 $a_1 \cdot f_1$, $a_2 \cdot f_2$, $a_2 \cdot f_2 f_1$ で使用した単語

生成手法	生成に使用した単語
$a_1 \cdot f_1$	方，品種，写真，着生，ラン，デンドロビウム，日本，茎，Dendrobium，種，属，商品，花，長生蘭，会，物，石斛，栽培，山野草，蘭，数，販売，富貴蘭，園芸，円，1，科，情報，鉢，ヤフオク，植物
$a_2 \cdot f_2$	方，the,（株）(注3)
$a_2 \cdot f_2 f_1$	方，日本，Dendrobium

$a_2 \cdot f_2$ ではセッコクに近い単語が得られていないため類似しない単語が出ていることが分かる．逆に $a_2 \cdot f_2 f_1$ では “Dendrobium” を選択できたため類似語に植物の学名として用いられる単語が現れている．また $a_2 \cdot f_2 f_1$ では $a_1 \cdot f_1$ の結果から分かる通り，与えられた単語集合がかなりよい形になっているため生成に失敗しなかったと考えられる．

4.2，4.3 の結果を踏まえ，Web 検索を用いたベクトル生成手法としては取得手法 s_1 では $a_1 \cdot f_1$ が，取得手法 s_2 では a_1, $a_1 \cdot f_3$, $a_2 \cdot f_2 f_1$ が 16 手法の中で精度の高いベクトルを生成できると判断する．

4.4　非学習済みの検索クエリに対する生成ベクトルの評価

提案手法を用いて，Word2Vec で学習を行えない複数単語の検索クエリ，自然言語による検索クエリに対してベクトルを生成し，得られる類似語が検索クエリの置換候補として適切かどうかを調査した．筆者が考案した 20 個の検索クエリについて，各類似語取得手法につき 5 つの類似語を取得し，得られた類似語が適切であるかどうかを被験者が評価する形で評価した．評価を行うにあたり，類似語取得手法の比較手法として検索結果に含まれる単語の中で最も出現回数の多かった単語を類似語とする tf と，Yahoo! JAPAN が提供するキーフレーズ抽出(注8) から得られるキーフレーズを類似語とする kp を採用する．実験では各クエリに対して，取得手法 s_1 において生成手法 $a_1 \cdot f_1$，比較手法 tf，kp を用い，取得手法 s_2 では生成手法 a_1，$a_1 \cdot f_3$，$a_2 \cdot f_2 f_1$，比較手法 tf，kp の 8 手法を用いて類似語を取得した．評価にあたり，クラウドワークス(注9) を利用し，一つの検索クエリに対し 5 人に評価をつけてもらい，類似語が適切であ

(注6)：（）は全角である
(注7)：(株) で 1 文字を表す環境依存文字
(注8)：http://developer.yahoo.co.jp/webapi/jlp/keyphrase/v1/extract.html
(注9)：https://crowdworks.jp/

表 11　各手法で得られる類似語の置換候補としての妥当性評価

妥当と判断した人数	s_1, tf	s_2, tf	s_1, kp	s_2, kp	s_1, $a_1 \cdot f_1$	s_2, a_1	s_2, $a_1 \cdot f_3$	s_2, $a_2 \cdot f_2 f_1$
5 人	13	15	15	14	8	9	9	3
3 人以上	25	28	31	29	16	20	17	12
1 人以上	43	51	59	54	37	39	39	30
0 人	57	49	41	46	63	61	61	70

表 12　“次期アメリカの大統領は？” に対し取得された類似語

比較手法 tf	比較手法 kp	提案手法 a_1	提案手法 $a_1 \cdot f_3$
取得手法 s_1	取得手法 s_1	取得手法 s_2	取得手法 s_2
次期	ドナルド・トランプ	大統領	米国大統領
大統領	ヒラリー・クリントン	オバマ	アメリカ大統領
トランプ	グローバー・ノーキスト	国務長官	前大統領
アメリカ	United States	アメリカ大統領	元大統領
氏	トランプ次期大統領	大統領候補	合衆国大統領

るかを調査した．

表 11 では被験者 5 人が各手法で得られた 100 単語に対し，妥当であると評価した人数に応じて分類を行った結果を示している．

提案手法内では，取得手法 s_2 における a_1 が最も妥当性のある単語を出しているという結果となった．また比較手法と見比べたとき，提案手法の精度は比較手法に劣る値となっている．このような結果となった要因として，検索クエリの置換候補が概ね一つに定まる検索クエリにおいては，検索エンジンの精度の高さから比較手法の推薦がうまくいきやすいこと，Wikipedia の情報を用いて学習を行うため，学習以降に起こった出来事に対して提案手法の推薦が非常に行いづらいことが挙げられる．

“次期アメリカの大統領は？” という検索クエリに対して，提案手法と比較手法を用いて取得した類似語を表 12 に示す．提案手法の学習データは 2016 年 6 月時点の Wikipedia の内容を用いているため，その当時大統領であった “オバマ” という単語が出現している．比較手法では 2017 年 1 月の検索結果から取得しているため “トランプ” という単語が出現するようになっている．

しかし検索クエリの置換候補が複数存在する場合において，比較手法と比べて提案手法の結果の精度がよいという傾向も見られた．表 13 では “京都 神社” に対する提案手法と比較手法によって得られた類似語を示している．このような結果となった要因として，複数の置換候補が存在する場合，一つ一つの候補の出現回数が相対的に減少することから，tf で取得しづらくなったこと，kp では，単語が検索結果に出現しない場合推薦ができないため，検索結果だけでは多くの置換候補が得づらいことが挙げられる．そのため，複数の置換候補を取得するという目的においては，比較手法となる tf, kp のように文章中に出現する単語を取得する手法よりも良い結果を得やすい傾向があると思われる．

5. まとめと今後の課題

本研究では検索クエリに対して，Web 検索と Word2Vec を用いて類似語を推薦することを提案した．実験を通して，1 単語に対してベクトルを生成した場合，約半分の単語は提案手法を用いて類似語を得ることができている．また学習できない複数単語，自然言語のクエリについても，提案手法を用いることで複数の置換候補が存在するクエリへの置換候補取得において優位性があることも確認した．今後は，現時点では行えていない，ログデータを用いた検索クエリ置換手法との精度の比較を行う予定である．そのうえで，提案手法が効果的となる検索クエリの特徴を調査していく．また，現在の手法についても，本論文では名詞を出すことを目的としているため，関連語も名詞のみを用いてベクトルを生成しているが，形容詞や動詞を含めることで精度が変わるかどうか調査を行っていく．形容詞のみ，動詞のみを使うことで形容詞や動詞が抽出できるかという点も今後の課題である．さらに，検索クエリそのものの分散表現を得て，ユーザが検索したいクエリの候補を提示する研究を Mitra [11] が行っている．Mitra の研究を基に，より適切なベクトル生成手法の考察や，Word2Vec 以外の分散表現取得手法を用いることを検討していく．

表 13　“京都 神社” に対し取得された類似語

比較手法 tf	比較手法 kp	提案手法 a_1	提案手法 $a_1 \cdot f_3$
取得手法 s_1	取得手法 s_1	取得手法 s_2	取得手法 s_2
京都	上賀茂神社	縁結び	伏見稲荷大社
神社	伏見稲荷代謝	伏見稲荷大社	八坂神社
紹介	ご利益	商売繁盛	北野天満宮
縁結び	京都霊山護国神社	厄除け	石清水八幡宮
寺	電話&FAX	下鴨神社	貴船神社
情報	京都人	八坂神社	下鴨神社
社	物集街道	鞍馬寺	祇園社
上賀茂神社	恋占い	ご利益	春日大社
神社	下鴨神社	貴船神社	牛頭天王
京都観光	貴布禰総本宮	北野天満宮	秋葉神社

謝　　辞

本研究の一部は，平成 28 年度科研費若手研究 (B)(課題番号：15K16091) によるものです．ここに記して謝意を表すものとします．

文　　献

[1] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*, pp. 3111–3119. 2013.

[2] Zellig S Harris. Distributional structure. *Word*, Vol. 10, No. 2-3, pp. 146–162, 1954.

[3] Qi He, Daxin Jiang, Zhen Liao, Steven C. H. Hoi, Kuiyu Chang, Ee-Peng Lim, and Hang Li. Web query recommendation via sequential query prediction. In *Proceedings of the 25th International Conference on Data Engineering, ICDE 2009*, pp. 1443–1454, 2009.

[4] Reiner Kraft, Farzin Maghoul, and Chi-Chao Chang. Y!q: contextual search at the point of inspiration. In *Proceedings of the 2005 ACM CIKM International Conference on Information and Knowledge Management*, pp. 816–823, 2005.

[5] 今井良太, 戸田浩之, 関口裕一郎. Web 検索サービスにおける多義的なクエリ推薦手法. 日本データベース学会論文誌, Vol. 9, No. 1, pp. 7–11, 2010.

[6] Qiaozhu Mei, Dengyong Zhou, and Kenneth Ward Church. Query suggestion using hitting time. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM 2008*, pp. 469–478, 2008.

[7] Steve Chien and Nicole Immorlica. Semantic similarity between search engine queries using temporal correlation. In *Proceedings of the 14th international conference on World Wide Web, WWW 2005*, pp. 2–11, 2005.

[8] Xuanhui Wang and ChengXiang Zhai. Mining term association patterns from search logs for effective query reformulation. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, CIKM '08, pp. 479–488, 2008.

[9] Fernando Diaz, Bhaskar Mitra, and Nick Craswell. Query expansion with locally-trained word embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, Volume 1: Long Papers*, 2016.

[10] Taku Kudo, Kaoru Yamamoto, and Yuji Matsumoto. Applying conditional random fields to japanese morphological analysis. In *Proceedings of EMNLP 2004*, pp. 230–237, 2004.

[11] Bhaskar Mitra. Exploring session context using distributed representations of queries and reformulations. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, pp. 3–12, 2015.