C-14

挟みこむ検索：与えられた 2 オブジェクトの補間オブジェクト発見

Finding Intermediate Objects between Two Examples

旭 直人†　　山本 岳洋†　　中村 聡史†　　田中 克己†
Naoto Asahi　　Takehiro Yamamoto　　Satoshi Nakamura　　Katsumi Tanaka

1．はじめに

近年，検索エンジンは情報を探すのに必要不可欠なツールになってきており，ユーザは様々な検索エンジンを用いて情報を獲得している．一般的に，ユーザが明確な検索意図を持っており，的確なクエリを作ることができれば，容易に求める情報を得ることができる．しかし，明確な意図を持っていても適切なクエリを作ることができなければ，情報を得ることができない．また，そもそもクエリをキーワードとして表現することが困難であるものも多々存在する．例えば，ユーザの持っている 2 つの本の間の難しさを持つ本を見つける，「桶狭間の戦い」と「本能寺の変」の間に起こった出来事を見つける，といったような検索では，ユーザの求める情報をクエリ化して検索することは難しい．これは，ユーザが知りたいものの名前を知らないために，直接その名前でクエリを作れないことと現在のウェブ検索エンジンが指定されたキーワードを文書が含むかどうかという基準で結果を返すことに起因している．そのため，情報を得るためには何度もクエリを工夫し，間接的に求めていく必要がある．こうしたあるものとあるものの間に存在するオブジェクトを発見するような検索は，従来の検索エンジンでは困難である．本研究の目的は 2 つのオブジェクトを与えた時にそれらの間に位置するような適切なオブジェクト（補間オブジェクト）を発見することである．

2．補間オブジェクト

補間オブジェクトとは，ユーザが与えた 2 つのオブジェクトのある軸上での間に位置する適切なオブジェクトであると定義する．軸とは，2 つの与えられたオブジェクトの間に何が来るべきかを決めるある観点のことである．例えば，2 つの入力が「京都」と「東京」であれば，位置の観点なら間に来るのは「名古屋」になる．入力が「クリントン」と「オバマ」で，大統領の順番という観点なら 2 人の間に来るのは「ブッシュ」になる．

2.1　システムデザイン

ユーザは，ある 2 つの間のものを求めるために，入力としてその 2 つのものをシステムにクエリとして与える．入力したキーワードに様々な意味がある場合，トピックを限定するためにトピックもクエリに加える．その結果，ユーザはシステムが求めた補間オブジェクトの結果を得ることができる．

システムはユーザからの 2 つの入力を受け取ると，システムは 2 つの入力に対する適切な並び替え軸を推定する．軸については 2.3 節で詳述する．軸推定後は，Web から補間オブジェクトの候補を取得し，フィルタリングにより候補から不適切なものを除外する．最後に，システムは推定した軸に従い候補を並べ，ユーザに結果を返す．（図 1）

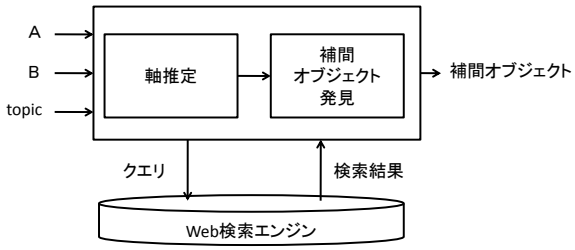

図 1.　システム概要

2.2　入力の種類

ユーザはクエリとしてキーワードを入力する．ユーザが入力するキーワードの種類には，人，グループ，組織，場所，イベント，概念，プロセス，階級など様々なものが考えられる．

ユーザによって入力される 2 つのキーワードは，何かしらの同じトピックに属しており，求めているものも同じトピックに属しているという前提で処理する．例えば，ユーザが「ブルース・リー」と「ジャッキー・チェン」を入力として与えた場合，ユーザはそれら 2 人の間に位置するカンフー映画スターを求めていると考えられる．

2.3　候補の並び替え

補間オブジェクトを見つけるために，ある軸上で集めた候補語を並べなくてはならない．並び替える軸にはユーザ依存の軸とユーザ非依存の軸の 2 種類がある．

ユーザ非依存の軸とは，そのクエリを入力したユーザに依存せず，入力に対して絶対的な値を決めることができるような軸である．例えば，本の値段や事件の起こった日時は絶対的なものであり，ユーザ非依存の軸であるといえる．

一方，ユーザ依存の軸とは，ユーザに依存して候補を並び替えるような軸である．例えば，ホテルを比較する時，その評価はユーザによって変わってしまうため，どのホテルが良いのかを絶対的に決めることはできない．このように一意に定まらない評価軸はユーザ依存の軸であるといえる．

2 つの軸の種類に加えて，軸は入力に依存するということを考慮しなければならない．2 つの入力が与えられた時，並べ替えに使う軸で多くの人が想定すると予想される軸が存在する．例えば，2 つの入力が「クリントン」と「オバマ」であれば，ほとんどの人は大統領の順番によって候補を並べるであろう．一方で，ユーザが 2 つの入力を入れたときに，考えられる軸が複数ある場合がある．例えば，2 つのホテルが入力として与えられた場合，ある人は

†京都大学大学院　情報学研究科　社会情報学専攻

値段で並び替えるかもしれないし，ある人は距離で並び替えるかもしれない．

本論文では，ユーザ非依存な軸と多くの人が並べ替えにひとつの軸を想定するような入力を扱う．これらの軸は得られた候補が適切かそうでないかが容易に判定できる．

3．検索エンジンを用いての補間オブジェクト発見

本論文では，検索エンジンを用いて補間オブジェクトを発見する手法を提案する．2.3 節で述べたように，本論文では，多くの人が 1 つの軸を想定するような入力の間にあるユーザ非依存の軸上のオブジェクトの発見に注力する．これらの種類の補間オブジェクトは，Web ページ上に記載される場合，多くの人がその順番を保ったまま記述すると考えられる．また，文章中で *A* の前や *B* の後ろに現れるような語は，そのトピックでの一般的な語である可能性が高い．そこでそうした語は適切な補間オブジェクトではない，と仮定をおいた．例えば，「桶狭間の戦い」，「長篠の戦い」，「本能寺の変」という 3 つのイベントを考えた場合，これらのイベントはこの順番でページに記載することが多いと考えられる．そして「信長」や「今川」といった言葉はページのいろいろな箇所で出現する．つまり，補間オブジェクトの抽出に当たり，2 つの入力に対する補間オブジェクトはウェブページ中に現れる 2 つの入力の記述の間に出現しやすいのではないかと考えられる．この仮定に基づき，補間オブジェクトの抽出を行う．実際の抽出の流れは以下の通りである．

1. システムはユーザの入力した 2 つのオブジェクトの間に現れる語を候補語として収集する．
2. システムは不適合な候補語を除くためにフィルタリングを行う．まず，2 つの入力の間に語が現れる頻度と入力の前方や後方に語が現れる頻度を比較して候補語のフィルタリングを行う．
3. 出現頻度が少ないものを除外する．
4. システムは残った候補語に対して，入力と共起しているかどうかに基づいてフィルタリングを行う．
5. 最後に，結果を語の頻度によってランキングし，ユーザに補間オブジェクトとして提示する．

3.1 候補語の収集

まず，システムは 2 つのオブジェクト名，*A* と *B*（ドメインを特定するのに必要であればトピック *T* も加える）をユーザからの入力として受け取る．ここで，問題を簡単にするために，ある軸上で *A* は *B* より小さいと仮定する．次に，システムはクエリ”(T∧)A∧B”を作り，検索エンジンにクエリを投げる．検索結果を得た後，システムは検索結果からスニペットを取り出す．システムはそれらのスニペットから *A* と *B* の間に出現する語を補間オブジェクトの候補語として収集する．ただし，収集の対象とする候補語の品詞は名詞と名詞句に限る．以降，収集した候補語集合を $C=\{t_1,t_2,...,t_n\}$ と置く．

3.2 語の出現頻度の比に基づく候補語のフィルタリング

3 章冒頭で述べた仮説に基づき，*A* と *B* の間に現れる語の出現頻度と *A* の前や *B* の後ろに現れる語の出現頻度の比を用いてそのような語のフィルタリングを行う．*C* に含まれるそれぞれの語 *t* に対し，以下のスコアを求める．

$$Frac_{(A,B)}(t)=\frac{tf_{between(A,B)}(t)}{tf_{between(A,B)}(t)+tf_{before(A)}(t)}\bullet\frac{tf_{between(A,B)}(t)}{tf_{between(A,B)}(t)+tf_{after(B)}(t)}$$

ただし，$tf_{between(A,B)}(t)\neq 0$

$tf_{between(A,B)}(t)$は *A* と *B* の間に現れる語 *t* の出現頻度である．$tf_{before(A)}(t)$は，*A* の前に出現する語 *t* の出現頻度，$tf_{after(B)}(t)$は，*B* の後に出現する語 *t* の出現頻度である．A の前や B の後に *t* が多く出現すればするほど，$Frac_{(A,B)}(t)$の値は小さくなる．*t* が *A* と *B* の間にだけ現れる場合，$Frac_{(A,B)}(t)$の値は 1.0 になる．システムは $Frac_{(t,A,B)}$の値が閾値αより小さければ語 *t* を一般的な語とみなし，候補から除去する．さらに，$tf_{between(A,B)}(t)$の値が非常に小さい場合も，語 *t* は補間オブジェクトとして不適切である可能性が高い．よって，$tf_{between(A,B)}(t)$の値が閾値β以下の語 *t* も除去する．候補語集合それぞれの語の出現頻度 *tf(t)*の中で最大の語の出現頻度 tf_{max} は入力により異なるので，βは以下のように定義する．

$$tf_{between(A,B)}(t)=\begin{cases} tf_{between(A,B)}(t) & (Frac_{(A,B)}(t)\geq\alpha)\wedge(tf_{between(A,B)}(t)\geq\beta) \\ 0 & otherwise \end{cases}$$

$$\beta=\lambda\cdot tf_{\max}\ (0\leq\lambda\leq 1)$$

ただし、$tf_{\max}=\max_{t\in C} tf(t)$

3.3 共起に基づく候補語のフィルタリング

前節で述べたように候補語をフィルタリングすることによって不適切な語をある程度取り除くことができる．しかし，残った候補語には入力のドメインに属していない不適切な語が依然として含まれている．そこで，A や B のドメインに属しているかどうかにより，候補語をフィルタリングする必要がある．このため，検索エンジンを用いて語の共起度合いを測る尺度である WebPMI を用いた．Bollegala[1]らは 2 つの語の間の意味的類似度を測るために WebPMI を用いている．WebPMI の定義は以下の通りである．

$$WebPMI(P,Q)=\begin{cases} 0 & if\ (P\cap Q)\leq c \\ \log_2\left(\dfrac{\frac{H(P\cap Q)}{N}}{\frac{H(P)}{N}\frac{H(Q)}{N}}\right) & otherwise. \end{cases}$$

*H(x)*は検索エンジンから返ってくるクエリ *x* のヒット数である．*N* は検索エンジンにインデックスされている全ページである．パラメータは $c=5$，$N=8*10^{10}$ に設定した．

A や *B* と同じドメインに *t* が含まれているかを判定するため，*WebPMI(t,A)*と *WebPMI(t,B)*の平均を用いた．平均値が *WebPMI(A,B)*の値に近い時，*t* は *A* や *B* と同じドメインに属している可能性が高い．

$$WebPMIFrac(t)=\frac{average(WebPMI(t,A),Web(t,B))}{WebPMI(A,B)}$$

t が同じドメインに属していれば，*WebPMIFrac(t)*は以下の条件を満たすと仮定し、以下の条件を満たさない語を候補語集合から除去する。

$$1.0-\gamma\leq WebPMIFrac(t)\leq 1.0+\gamma$$

表 1. 実験で用いたクエリ

カテゴリ	サブカテゴリ	正解セット数	生成したクエリ数	入力例	出力例
人物	徳川将軍家	15	91	(家康, 吉宗)	家光
	天皇	10	36	(神武天皇,崇神天皇)	開化天皇
出来事	戦国時代	15	3	(桶狭間の戦い, 本能寺の変)	長篠の戦い
	第 2 次世界大戦	10	3	(満州事変, ポツダム宣言)	ミッドウェー海戦
出世魚	ブリ	6	10	(ワカナゴ,ブリ)	ハマチ
	ボラ	6	10	(ハク, トド)	スバシリ
場所	京都の通り	24	276	(御池, 五条)	四条

表 2. 候補語集合の上位 k 件(@k)が 1 つでも正解を含む割合

	全サブカテゴリ	徳川将軍	天皇	戦国時代	第 2 次世界大戦	ブリ	ボラ	京都の通り
@1 (%)	82	86	78	66	100	10	40	86
@3 (%)	85	90	80	100	100	10	40	90
@5 (%)	86	90	80	100	100	10	40	90

表 3. それぞれのフィルタにおける正棄却率と誤棄却率

		全サブカテゴリ	徳川将軍	天皇	戦国時代	第 2 次世界大戦	ブリ	ボラ	京都の通り
TF-比	正棄却率 (%)	18.68	15.05	18.44	18,68	13.87	24.48	27.67	18.10
	誤棄却率(%)	13.09	3.50	3.41	0.00	22.22	51.67	10.83	1.83
TF-カット	正棄却率(%)	84.95	99.57	99.70	98.28	99.67	97.89	99.53	98.53
	誤棄却率(%)	26.15	30.46	16.73	33.33	33.33	45.00	24.17	21.13
共起	正棄却率 (%)	9.52	0.00	0.00	66.67	0.00	0.00	0.00	0.00
	誤棄却率(%)	10.00	0.00	0.00	0.00	0.00	0.00	70.00	0.00

3.4 候補語のランキング

上記の手法を用いて候補語をフィルタリングすることによって，最終的な候補語集合を得ることができる．今回は，最終的に残った候補語集合内の語を出現頻度の降順でランキングし，その結果をユーザに提示する．

4. 実験

提案手法により，補間オブジェクトがどの程度の精度で抽出できるのかを評価するために実験を行った．実験で用いたクエリは表 1 に示している．出来事，人物，出世魚，場所の 4 つのカテゴリを設定し，それぞれのサブカテゴリに対し，正解セットを用意した．正解セットに含まれるオブジェクトは予め順番が決まっている．例えば，サブカテゴリ「徳川将軍」の正解セットでは，「家康」，「秀忠」，「家光」，…，「慶喜」と予め一意に順番が決まっている．これらの正解セットの中から最低でも間に 1 つの補間オブジェクトがあるようにしながら 2 つ選び，クエリを作る．そして，生成されたクエリそれぞれに対し手法を適用し，それぞれ結果を得た．得られた結果は，正解セットに含まれていてかつクエリとして与えられた 2 つの要素の間に位置するものである時，正解であるとした．全ての結果を得た後，それぞれのサブカテゴリで上位 k 件に少なくとも 1 つの正解が含まれる割合を求めた．その結果，表 2 のようになった．

また，3 章で述べたそれぞれのフィルタがどの程度の効果があるか測定を行った．これらのフィルタを評価するために 2 種類の値を測定した．1 つはシステムが収集した全ての候補語の中に含まれる不適切な候補語すべてからどれだけ不適切な語を除けたかという割合であり，もう 1 つは候補語の中に含まれる正しい候補語を誤って除去してしまった割合である．前者を「正棄却率」，後者を「誤棄却率」と呼ぶことにする．正棄却率と誤棄却率を語の出現頻度の比を用いたフィルタ（a だけを適用した場合のもの（TF-比）と，最大出現頻度を用いたフィルタだけを適用した場合のもの(TF-カット)を求める．2 つのフィルタを適用した後に共起を用いたフィルタを適用した場合（共起）の正棄却率と誤棄却率を求める．この場合，2 つのフィルタによって残っている語の中から計算する．結果は表 3 のようになった．それぞれのパラメータは経験的に$\alpha = 0.6$, $\lambda = 0.5$, $\gamma = 0.2$ とした．検索エンジンは Yahoo! Web Search を用い，上位 200 件の結果を取得した．

表 2 より，提案手法は平均して上位 1 件で 80%以上の精度を達成している．表 3 より，最大出現頻度を用いたフィルタではうまくいっている例では約 99%の不適合な候補

語を除去できているが，それらでは同時に適合した候補語まで除いてしまっている．ほとんどの場合，得られた候補語集合の中に少なくとも 1 つは正解を含んでいたが，フィルタリングが強力すぎて全ての正解を除去してしまっている場合があった．これには，クエリが多くの人が記述しそうなメジャーなもので，候補語があまり多くの人が記述しそうにないマイナーなものである時に顕著に起こっているように思われた．メジャーなもの同士の共起度とメジャーなものとマイナーなものの共起度は異なると考えられるので，今後メジャー，マイナーの関係を考慮していく必要があると考えられる．また，出世魚の「ブリ」と「ボラ」のサブカテゴリでは特に結果が良くなかったが，これは正解セットを 1 種類しか用意しなかったが実際には地域によって呼び名が異なってしまうことに起因すると考えられる．地域による呼び名の違いを考慮すれば精度は上がるものと思われる．

この実験を通し，フィルタで不適合な候補語を取り除くだけでなく，正解である候補語をうまく上位に持ってくることが重要であると分かった．今後，フィルタリングとランキングをうまく組み合わせた手法を考えていきたい．また，今回はある軸上の順番とテキスト中の出現順序が一致するという前提で補間オブジェクトの抽出を行ったが，それがどの軸（時間軸や空間軸など）に分類できるのか，という推定を行っていない．言語パターンや与えた入力と候補語の距離などを考慮に加え、テキスト中の出現順序に意味がある文脈なのか，そうでないのかを判断し，またそれがどの軸に分類できるのか，といったことを今後考えていく必要がある．

5. 関連研究

Gang ら[2]は人物，場所，会社といったようなカテゴリで 2 つのエンティティ間の関係を発見する手法を提案している．まずユーザに 2 つのクエリを入力してもらい，Web からそのクエリ間の関係を発見し，類似度の降順でページの組をユーザに提示する．Sun ら[3]は 2 つの検索結果のリストを比べるのを手助けする CWS システムを提案している．CWS システムはユーザから 2 つのクエリを受け取り，2 つの検索結果の組み合わせにより 2 つの結果の並び替えを行う．我々の提案システムは 2 つのクエリを受け取るという点で似ているが，彼らの手法が 2 つのクエリ間の関係性を発見することが目的であるのに対し，ある基準で補間オブジェクトを発見しようとする点が大きく異なる．木村ら[4]は，Web から人物の年表を生成するシステムを提案している．そのシステムでは，ユーザから人物名を受け取り，Web から自動的にその人物に関わるイベントを抽出し，年表の形にしてユーザに提示する．郡ら[5]はブログから主要な観光ルートを発見する手法を提案している．そのシステムでは，観光地を訪れた人のブログからその人が訪れた一連の場所のパターンを抽出し，PrefixSpan[6]を用いてルートの集約を行う．本研究は，補間オブジェクトを探すことを目的としている点でこれらの研究とは異なるが，年表を埋める，2 つの地点のベストなルートを探す，という手法は本研究に活用できると考えている．

6. まとめ

本論文では Web 検索エンジンを用いて 2 つの入力の補間オブジェクトを発見する手法の提案を行った．提案手法では，まず候補語を収集し，位置による出現頻度の比と最大出現頻度を用いたフィルタで不適合な候補語を除き，その後に WebPMI を用いた共起度を見ることでさらに不適合な候補語を除く．最後に出現頻度でランキングを行い，ユーザに提示する．

実験では 80%程度の精度を示した．加えてメジャーであるか，マイナーであるかを考慮に加えなければならないこと，除去を強調するのではなく，正解を上位に上げてくるアルゴリズムに改良することが必要であることがわかった．

今回は適切な補間オブジェクトを 2 つの入力と同じドメインに属し，2 つの入力の間に最も現れたものとしたが，場合によっては 2 つの入力の平均をとるものが適切な補間オブジェクト（例えば 2 つの本の値段の平均の値段を持つ本）であるかもしれない．適切な補間オブジェクトとは何かということを今後考えていく必要がある．また，容易に想定できる軸が 1 つであり，一意に正解が決まるようなものを対象として扱ったが，今後は軸が複数想定でき，正解が一意に定まらないようなものも扱っていきたいと考えている．また，2 つのオブジェクトの内側だけでなく，外側に存在するオブジェクト（大正、明治なら昭和）も発見できるように手法の拡張を考えていきたい．

謝辞　本研究の一部は，グローバル COE 拠点形成プログラム“知識循環社会のための情報学教育研究拠点”，計画研究　“情報爆発時代に対応するコンテンツ融合と操作環境融合に関する研究”(研究代表者：田中克己，A01-00-02, 課題番号 18049041)，未踏 IT 人材発掘・育成事業 2009 年度上期　未踏ユースによるものです．ここに記して謝意を表すものとします．

文　献

[1] D. Bollegala, Y. Matsuo and M. Ishizuka. “Measuring Semantic Similarity between Words Using Web Search Engines”, Proceedings of the 16th International World Wide Web Conference (WWW 2007), pp. 757-766 (2007).

[2] Luo, G, Tang, C and Tian Y. “Answering relationship queries on the web”, In Proceedings of the 16th international conference on World Wide Web, pp. 561-571, 2007.

[3] J. Sun, X. Wang, D. Shen, H. Zeng, and Z. Chen. “CWS: A Comparative Web Search System”, In Proceedings of the 15th International Conference on World Wide Web, pp. 467-476, 2006.

[4] R. Kimura, S. Oyama, H. Toda and K, Tanaka. “Creating Personal Histories from the Web Using Namesake Disambiguation and Event Extraction”, In Proceedings of the ICWE2007, pp. 400-414, 2007.

[5] H. Kori, S. Hattori, T. Tezuka and K. Tanaka. “Automatic Generation of Multimedia Tour Guide from Local Blogs,” In Proceedings of the 13th International MultiMedia Modeling Conference, pp. 690-699, 2007.

[6] Pei. J, Han. J, Mortazavi-Asl, B. Pinto, H. Chen, Q. Dayal, U. and Hsu, M. “PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern”, IEEE Int. Conference on Data Engineering, 2001.