

UT-Kiwi: 検索支援としてのテキストマイニングシステム

吉田 稔† 中川 裕志†

東京大学情報基盤センター†

1. はじめに

近年、WWW 上や組織内に蓄積される電子的文書の量は増大の一途を辿り、それらの文書を人間が把握することが困難となっている。WWW 全体における文書量の増大のみならず、その中でのトピックの限定された部分集合（特定サイト内の Web 文書集合、Wikipedia、さらには企業内文書集合等）のサイズも増大し、WWW 全体と同様に、把握が困難となりつつある。

検索エンジンは、文書集合を効率的に利用する有力な手段の一つであるが、その機能は基本的には「入力された文字列（クエリ）を含む文書を返す」という範囲に留まっており、「どのようなクエリを入力するべきか」という問題に対しては、研究の余地が多く残されている。

本研究では、このような「クエリ入力支援」に対する新しい提案として、テキストマイニングによる検索支援システム UT-Kiwi を提案する。

テキストマイニングとは、与えられたテキスト集合の中での、「言葉の使われ方」（主に、言葉に関する統計的情報）について分析するタスクである。UT-Kiwi では、入力されたクエリに対し、「用例抽出」「同義語抽出」という二種類のテキストマイニングをリアルタイムに行い、マイニング結果を提示する。

UT-Kiwi は、前述の「トピックの限定された文書集合」を主に対象としており、それらのテキストに対する検索を支援する。基本アイデアは、「文書集合をオンメモリで配置し、Suffix Array による高速検索を実装する。さらに、Suffix Array による検索を応用することで、高速なテキストマイニングを行う」というものである。これにより、テキストマイニングを検索支援として用いることが可能となる。

2. システム概要

UT-Kiwi*は、現在東京大学の Web サイト内の HTML 文書を文書集合として、サービスをを行っている。図 1 に、UT-Kiwi の画面を示す。図中、A. に示されているのが検索窓であり、ユーザがここにクエリを入力すると、システムは入力内容に応じて用例や同義語を提示する。ここで用例とは、クエリの前方、後方にどのような文字列が頻繁に接続しているかを示したものであり（図中「B. 後方接続」「C. 前方接続」）、この場合、「シンポジウム」の前方に「記念」や「国際」、後方に「日本」や「講演」といった言葉が続き易いことがわかる。同様に、シンポジウムの同義語として、「講演会」や「ワークショップ」という文字列が用いられやすいということも提示する（図中、「D. 類義語・同義語」）。ユーザは、提示された用例等をクリックすることで、クエリの補完を行える。補完された結果は、そのまま Google へのクエリとして、検索に用いることができる。



図 1 : UT-Kiwi*

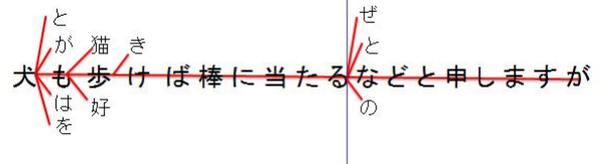


図 2 : Kiwi アルゴリズム概要

3. テキストマイニングの詳細

3.1. 用例検索

用例検索は、Kiwi というアルゴリズムを用いて行う。Kiwi についての詳細は、文献[1]を参照されたい。Kiwi では、クエリを検索エンジンで検索し、その結果をマイニングに用いていたが、UT-Kiwi では、これを Suffix Array によるオンメモリの検索に置き換えることで、応答時間を大幅に短縮し、クエリ補完に利用可能な速度を実現している。

図 2 に、Kiwi アルゴリズムの概要を示す。Kiwi アルゴリズムでは、ある文字列 s が与えられたとき、「 s に接続する文字の種類数」（以下、「分岐数」）を計測する。例えば、図 2 では、「犬も」の後続文字として、「猫」「歩」「好」の 3 種類の文字があり、分岐数は 3 となる。文字列 s にどのような文字列が接続するかを、木構造の探索により列挙していくが、そのさい、分岐数が増加している場合のみ、用例候補として考慮する。（図 2 の例では、例えば「犬も歩けば棒に当たる」が、分岐数 1 から 4 へ変化する点として、候補になる。）これは、「分岐数の増加は、意味的な切れ目とよく合致する」という経験則に基づいたルールである。また、UT-Kiwi では、用例候補が見つかったノードに対しては、それ以上深く探索を行わないという制限を加えることにより、探索の高速化を行っている。

最終的に、このようにして得られた用例候補を、スコア関数

$$\text{score}(s) = \text{freq}(s) \cdot \log(1 + \text{length}(s))$$

により順位付けし、提示する。

3.2. 同義語抽出

同義語抽出において問題となるのは、ユーザーによって入力される文字列の多様性である。同義語抽出や言い換えに関しては従来より多くの研究が行われているが、

UT-Kiwi: A Text-Mining-Based Query Support System

† Minoru Yoshida and Hiroshi Nakagawa, Information Technology Center, the University of Tokyo.

* <http://kiwi.r.dl.itc.u-tokyo.ac.jp/ut-kiwi/>

その大部分は、予め規定された候補語（例えば、「名詞」や「動詞+目的語」）どうしの類似度を測るものである。このため、本システムが想定する、「ユーザがあらゆる文字列を入力する状況」においては、文書のあらゆる部分文字列どうしについての類似度を計算する必要が生じ、必要な計算時間および記憶容量は膨大なものとなる。この問題に対し、本研究では、「ユーザからクエリが与えられた際、On-Demand にそれと類似する文字列を文書から取り出す」という問題設定を考える。このさいに要求されるのは、「どんな文字列にも対応できる」「高速な」アルゴリズムである。この実現に、Suffix Array による検索を活用する。以下、アルゴリズムの概要について解説する。

3. 2. 1. アルゴリズム

同義語抽出アルゴリズムでは、一般に「意味の類似する文字列は、同一の文脈で用いられることが多い」という仮説に基づき、文脈の類似度を計算することにより文字列の類似度を測定する。特に、「隣接する文字列」は、文脈として非常に有用であることが報告されており[2]、この「隣接する文字列」は、Suffix Array により容易に検索することができる。このため、本研究では、(1)「クエリ文字列に隣接する文字列」を検索し、(2)得られた隣接文字列から適切な「文脈」を選択し、(3)「文脈に隣接する文字列」を検索するという3段階の処理により同義語を抽出するアルゴリズムを開発した。以下、上記(1)(2)の処理をSTEP-1とし、(3)をSTEP-2として説明を行う。

高速化のため、文字列の検索を探索木として表現（図3・図4）し、探索対象の文字列のスコア付けを探索と同時にアルゴリズムを採用した。これにより、枝狩りによる探索空間の削減が可能になった。

以下、簡単のため、右側接続文脈の探索についてのみ述べる。STEP-1では、クエリ q に接続する文脈 x のスコアを

$$\text{score}(x) = \text{freq}(q.x) \log |S| / \text{freq}(x)$$

で定義し（freq は頻度、q.x は文字列 q と x の連結、|S| はコーパス全体の文字列長）、スコア上位 N 個の文脈を取得するが、その際に、

$$\text{score}'(x) = \text{freq}(q.x) \log |S|$$

なる上限を用いて枝狩りを行う。 $\text{score}'(x) \geq \text{score}(x.y)$ が如何なる y についても必ず成り立つため、 $\text{score}'(x)$ が現在 N 位の候補のスコアを下回れば、それ以上の探索を打ち切ることができる。

STEP-2では、効率的な枝狩りを行うため、(a)枝狩りに有用なスコア関数により、候補語を絞り(STAGE-1)、(b)より精緻なスコア関数により、絞られた候補語のランキングを行う(STAGE-2)、という2段階の処理を行う。

STAGE-1では、「同義語候補 x の左側（右側）に接続する文脈の種類」を「左（右）スコア」と定義し、まず左スコア上位 N 個の同義語候補、続いて右スコア上位 N 個の同義語候補を探索する。左スコア、右スコア共に、探索の深さに対する非増加関数となるため、現在のスコアが N 位のスコアを下回った場合、そこで探索を打ち切ることができる。こうして得られた 2N 個の候補を、最終的に、スコア関数

$$\text{score}(x) = \sum \log(\text{freq}(c+x) / \text{freq}'(c+x))$$

により順位づけする。ここで、 Σ は、すべての文脈に関する和、 $c+x$ は、文脈 c と文字列 x を適切な順序で連結したものを表し、 freq' は、 $\text{freq}(x) \cdot (\text{freq}(c) / |S|)$ で定義される値であり、「c と x が無関係だった場合の x の予測頻度」をモデル化した値である。

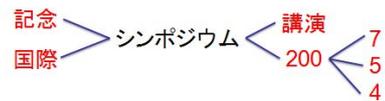


図3：STEP-1 概要（文字列「シンポジウム」に接続する文字列の探索木の例）

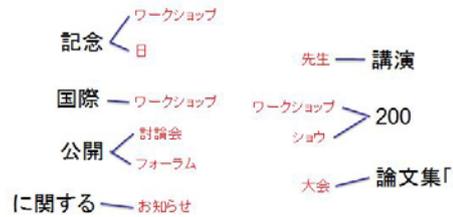


図4：STEP-2 概要

(各文脈に接続する文字列の探索木の例)

4. 実験

本研究で提案する同義語抽出アルゴリズムに関し、実験を行った。実験は航空分野のレポート（7Mbytes）と、それに付随する同義語辞書を用い、「辞書に登録された語に対し、同義語を検索する」という問題設定により行った。Average Precision を指標として用いたところ、従来一般的に用いられる、「周辺単語」と「構文構造」を用いる同義語抽出手法(VSM)の精度は、複数の重みづけ手法により実験したところ、最高で 57.35 ポイント（最低で 23.25 ポイント）だったのに対し、提案手法の精度は 52.20 ポイントであり（表1）、従来手法には及ばないものの、最高精度に近い精度を得ることができた。また、1クエリあたりの応答時間は、約2秒であった。

表1：実験結果

アルゴリズム (重みづけ手法)			
提案手法	52.20		
	構文構造	周辺単語	組み合わせ
VSM(logTF)	34.43	55.61	57.35
VSM(TFIDF)	24.72	37.85	39.19
VSM(TF)	23.25	40.12	40.87

5. おわりに

テキストマイニングに基づく検索支援システム UT-Kiwi を提案した。UT-Kiwi では、クエリ入力に対し、文書集合から「用例抽出」「同義語抽出」を On-Demand にを行い、ユーザに提示する。提案アルゴリズムは、任意のクエリ入力に対し用例・同義語を抽出できることが特徴である。7Mbytes のコーパスに対する同義語抽出実験では、1クエリ当たりの応答時間は約2秒であり、抽出精度は、一般的に用いられるアルゴリズムと比較し、やや劣る値であった。今後は、より大規模なコーパスでの実験および高速化を図る予定である。

参考文献

- [1] K.Tanaka-Ishii, H.Nakagawa, A Multilingual Usage Consultation Tool based on Internet Searching ---More than search engine, Less than QA, WWW2005
- [2] M.Hagiwara, Y.Ogawa, K.Toyama. Selection of Effective Contextual Information for Automatic Synonym Acquisition. COLING/ACL 2006