トラックバックを利用したブログ記事間の関連性の抽出

Extraction of the Relations between Blog Articles using Track Back

新美　礼彦 1)　　武山　弘樹 1)　　小西　修 1)

Ayahiko Niimi　　Hiroki Takeyama　　Osamu Konishi

1) 公立はこだて未来大学 システム情報科学部

School of Systems Information Science, Future University-Hakodate

Abstract: In this paper, we proposed the system that extracts only the track back to the blog related to the content of the original article from among the track back, and developed the system as extension of Firefox. The flow of this system is as follow. The proposed system traces the track backs, analyzes the original article and the track back articles, calculates the evaluation value of each article, and judges the tarack back relations by evaluation values. To evaluate the proposed system, we applied it to various blogs, and it was confirmed to be able to extract the track backs with the relation as a result of the experiment. Especially, the track back spam was able to be filtered. Because it is difficult to judge the relation between the article and the article.we extracted only storongly-relation in the our experiment, so the proposed system sometimes filtered the article wanted to read by the user. We will improve the extraction algorithm in the future.

1　背景と目的

最近ではブログが人気を集めている。総務省の調べによると平成 18 年 3 月末の時点で 868 万人ものブログ登録者数がいる。[1] これは個人で手軽に利用できる点などがうけ、気軽に個人が情報発信できるということや、多くの著名人がブログを書いていることなどもブログ登録者数の増加に影響していると考えられる。そのため現在は多くのブログが存在しており、人気のあるブログには数百件のトラックバックがつくこともある。

ブログを見ている時に、関連する記事を書いている他のブログに行きたくなる時がある。その際、検索サイトやトラックバックなどを使うことが多い。しかし、検索サイトを使う場合は、ブログを一から自分で探さなければいけないため、探すのが大変である。また、トラックバックを使う場合でも、トラックバックスパムや、すでに削除された記事への無効なリンクがある場合などが多く、探すのが大変である。

本論文では、トラックバックの情報を使い、そこから必要な部分だけ取り出せれば、元の記事と同じテーマについて書かれているブログを探せるのではないかと考えた。本論文の目的は、トラックバックの中から元記事と記事の内容に関連のあるブログを抽出することとする。

2　関連研究

ここでは本研究に関連のあると思われる技術や研究などについて述べる。

最近は一般生活やテレビ番組などで “ブログ” という言葉をよく聞くようになってきた。ブログの名前の由来は “Web” と “Log” を組み合わせて作った “Weblog” という言葉である。現在はそれを省略してブログと呼ぶのが一般的となっている。その名があらわしているように、もともとはウェブ上のウェブページの URL と共に、メモや論評を書き足して記録していくウェブサイトの一種であった。

ブログの用途は広く、人によって使い方は様々であるが、よくあるブログの形態としては、日々の出来事を日記のように綴っていくものや、毎回ひとつの記事を探し、そこへのリンクを張り、一言コメントするようなタイプのものなどがある。

ブログに似たものに、Web 日記が挙げられる。Web 日記はブログサービスやブログツールなどを使わずに HTML を編集するなどして、Web 上に自分の日記を載せるものである。ブログも自分の日記を載せること

ができる点では同じだが、Web 日記ではコメントやトラックバックをつけることができない。日記を書いた時点で完結してしまうという点が Web 日記とブログの違いだといえる。

ブログは、1 日ごとまたは 1 日を複数個の「エントリ」と呼ばれる個別書き込み記事の集合からなっている。多くのブログでは、1 エントリで 1 トピックスを扱っているものが多い。

2.1 ブログの機能

現在では必ずしもブログだけで使われているわけではないが、ブログには他のサイトでは使われていなかったいくつかの技術が使われている。ここでは、それらのブログ特有の機能について説明をする。

ブログサービス ブログサービスとはブログを公開するための Web サーバー等を、自分で用意する必要がなく、申し込むだけで簡単にブログを作れるものである。現在ブログをつくるためのサービスは数多くある。ブログの普及に努めている日本ブログ協会 [2] によると、エキサイトブログやココログ等、62 ものブログサービス提供者が存在し、それぞれブログサービスを提供している。このようなサービスでは、ブログサイトのテンプレートが何種類か用意されていて、そこから自分の好きなレイアウトのものを選ぶといったところが多い。

トラックバック トラックバックとは、ブログでよく使われている機能の一つで、ブログにリンクしたことを相手のブログに通知し、また相手から自分への逆リンクを自動的に生成するしくみのことである。(図 1 参照)

トラックバックスパム 最近では自分のブログへのリンクを増やすために、関係のないブログへトラックバックを行うケースが多くなっている。このようなトラックバックは、一般的にトラックバックスパムと呼ばれている。トラックバックスパムとは、ブログ記事とは無関係に行う迷惑なトラックバックのことである。アダルト系や出会い系、ワンクリック詐欺などのサイトに誘導するものや、アフェリエイト目当てのブログに誘導するものも少なくない。

コメント ブログにある機能の一つで、記事を読んだ人がそれに対し意見を言うためのものである。有名人のブログなどでは、いろいろなコメントが多く付くため、コメント機能そのものを使えなくしてしまっていたり、一度管理者側で確認してから選別したコメントを掲載するようにしているものも多い。

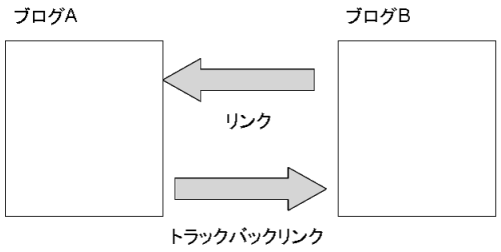

図 1: トラックバックの例

本研究ではトラックバックに注目して、ブログ間の関係をユーザに提示するシステムを提案する。コメント機能は、トラックバックが多い有名なサイトほどコメント機能が使えないサイトが多く、今回は使わなかった。

2.2 ブログに関する研究

ブログに関する研究はいくつもなされているので、関係の深いと思われるものについて以下に記す。

1. blogWatcher
   ブログを収集・監視し、集めたブログをマイニングすることによって、キーワードがいつ多く取り上げられたのかや、いっしょに使われるキーワード、ブログ作成者の性別など、様々な情報を閲覧することができるようにしたシステムである。 [3]

2. Blog Keyword Visualizer
   ブログで盛り上がっているキーワードの出現頻度や、キーワード同士の繋がり、またジャンル分けなども行いそれらの情報をアニメーションによって表示するソフトである。 [4]

3. トラックバックネットワークに基づく SEO コミュニティの分析
   この研究ではトラックバックで繋がっているブログを収集し、その中から SEO コンテストに関係する活動を行っているブログを発見し、分析するというものである。発見のためには、参加者がブログ中に特定のキーワードを利用する、SEO コンテストという特別な環境を実験に使っている。 [5]

4. Blog コミュニティの抽出と分析
   特定のキーワードに関するブログのリンクを収集し、それらのリンクの重要度を決めておき、ブログ間のリンク構造を解析し、弱いリンクを削除することによってコミュニティの抽出と分析を行ったものである。 [6]

1, 2 の研究ではあるキーワードに対して、ブログユーザーがもっているイメージなどは把握しやすく、特定のキーワードを使っているブログを探すのにはとても便利であるが、一度ブログを見つければそれで終わりとなってしまうという点がある。本研究では、気になるブログを見つけたときにそこから似たテーマについて話しているブログを探しやすいという利点がある。

3 の研究では、特定のキーワードを含むブログを対象としているが、記事の内容までは見ていない、これでは SEO コンテストなどの特別な状況下でしか使うことができない、本研究では記事の内容を使うことによって様々なブログでの関連抽出を目指している。

4 の研究では、ブログ間のリンクを元にコミュニティを抽出しているので、話題の種類がいくつもあるブログの場合は適切なコミュニティを見つけるのが難しいと考えられる。そこで本研究では記事に注目し、その記事の文章をもとに、記事間の関連を抽出するものである。

3 提案するシステム

本研究では、起点となるブログ記事の URL をユーザーに指定してもらい、そこからトラックバックリンクで繋がっているブログ記事に対して、起点となったブログ記事と関連がある記事なのかどうかという情報を、ユーザーに提供するシステムを作成する。

3.1 システム概要

本論文で構築するシステムは、主に Firefox の拡張機能 (Extension) として構築している。その理由としては、解析対象がブログということもあり、Web ブラウザとの連携が容易な方がいいと考えたからである。最終的にはブログを見ているときに気軽に使えるようなシステムになることを目指している。

Mozilla Firefox は Mozilla Fundation が 2004 年に発表したブラウザで、発表以来、急激にシェアを伸ばしているブラウザである。 [7, 8] このブラウザの特徴としてオープンソースである、クロスプラットフォームの実現している、非常に優れた拡張性を持つ、がある。

デザインと GUI 定義が分かれており、動作は Javascript によって拡張可能であることから、ユーザごとのインタフェースやデザインを簡単に実現することが可能である。Firefox では、Extension としてオリジナルにない機能を追加することが可能となっており、様々な拡張機能が公開されている。 [9, 10]

また、Firefox の拡張機能だけでは実現しにくい部分として、本文の解析部分があるが、形態素解析する部分については、外部のソフトを利用する。形態素解析には安定性、速度を考えて MeCab [11] を利用した。

システム全体の流れは、図 2 のようになる。

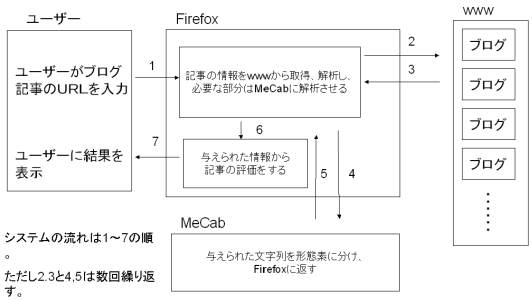

図 2: システム全体の流れ

3.2 アルゴリズム

提案システムでは、ブログ記事を指定し、そこにトラックバックをしているすべてのブログ記事に対して、元の記事との関連があるか無いかを判断し、関連のあるブログ記事の一覧をユーザに提示するシステムを検討した。システムのアルゴリズムを以下に示す。

1. ブログの種類を判断
   ユーザが解析したいブログ記事の URL を入力し、その URL からブログサービスを判断する。

2. 本文、トラックバック情報を取得
   ブログサービスごとに定義したアルゴリズムで、そのブログ記事の内容を解析し、本文とトラックバック一覧を取得する。

3. トラックバック一覧にある URL それぞれに対し、内容を取得、解析
   トラックバック一覧の URL それぞれに対し、1,2 と同じ内容を繰り返す。

4. 記事ごとに評価値を計算
   今までに得た情報を総合して、各記事に評価値を与える。

5. 結果の表示評価値が一定以上なら関連ありと判断し、一定以下なら関連なしと判断し、表示する。

まず、指定した URL から、本文とトラックバックを取得する必要がある。ブログはブログサービスごとにレイアウトが違うので、それぞれのブログサービスごとに本文、およびトラックバックを取得できるようにテンプレートを用意した。対応するブログサービスは、いくつかのブログランキングを元に、上位にランキングしているブログサービス 9 個を選んだ。以下に対応しているブログサービスのリストを示す。

1. ココログ
2. FC2 BLOG
3. Seesaa ブログ
4. Ameba ブログ
5. ヤプログ！
6. エキサイトブログ
7. 楽天ブログ
8. So-net blog
9. プチモールブログ

本文テキストを形態素解析ツール MeCab に渡し、形態素解析した結果を受け取る。形態素解析した結果から、名詞だけを取り出し、それぞれの単語が何個含まれているかを調べる。すべての記事を解析し、それぞれの単語に対して TFIDF により単語の重要度を計算する。単語の重要度を元に、記事と記事の関連の有無を評価する。TFIDF で求めた単語の重要度を a とし、その単語が含まれている数を b とする。$a \times b$ をすべての単語文だけ計算し、その和を x とする。記事の評価値 y は $y = x/n$(ただし、n はその記事の単語数) とする。

TFIDF により単語ごとの重要度を求め、記事中の単語数を考慮することにより、極端に長い記事 (すなわち、単語を多く含む) の評価値が大きくなることを防いでいる。

評価値に対する関連あり・なしの閾値は実験により求めた。

3.3 実装

本システムの使用方法の流れと、スクリーンショット (図 3) を以下に示す。

1. テキストエリアに解析したいブログ記事の URL を入力する
   アクティブなウィンドウの解析をしたい場合はツールを押し、URL 取得ボタンを押すことによって URL 入力の手間が省ける。
2. 解析ボタンを押す
   横のテキストエリアに数値を入れてから解析ボタンを押すことによって、解析するトラックバックの上限を決めることもできる。
3. 解析結果が表示される
   上段に関連があると思われるブログの URL、下段に関連がないと思われるブログの URL が表示される。

実際の作業としては、URL を入力して解析ボタンを押すだけであるが、その後結果が表示されるまでは時間が多少かかる場合がある。解析にかかるおおよその時間は、10 個を解析する場合は、1 分程度で、100 個になると 7 分ほどになる。

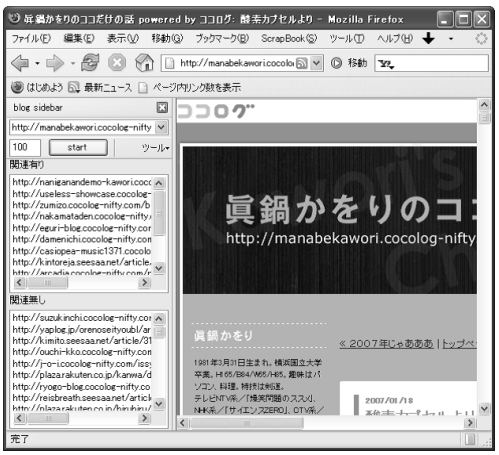

図 3: 作成したシステム

4 実験と評価

ここでは、システムを作るに当たって行った実験、および評価のための実験について述べる。

4.1 評価を決定するための実験

当初、名詞に重みをつけずに、記事と記事の比較を行った結果、単純に文が長ければ、評価が高くなる傾向が強くなり、まったく別の話題についての記事でも評価を高くしてしまう結果になった。

そこで提案のように、全ての名詞に重みをつけ、その記事を特徴づけるような名詞に高い評価を与え、それを使えばよいのではないかと考え、その検証をする実験を行い、その有効性を検証した。また、求めた評価値に対して、手作業でブログの内容を確認し、どのくらいの値だと実際に関連があるかを調べ、閾値を決定した。本実験では、提案システムから関連性の有無の判断機能だけを取り外し、ブログ記事を指定すればそこからトラックバックでリンクされる先の本文と元記事を比較して評価値を出力するシステムを用いた。

実験の結果、関連のない記事は評価値が低くなりやすいことが確認できた。関連がない記事は評価値が 0.1 未満になることが多く、関連がある記事は 0.1 以上になることが多かったため、以後、0.1 を閾値として関連性の有無を判断することにした。

4.2 関連ある記事の抽出確認

提案システムで実際にどの程度関連記事を抽出できるのか、様々なブログで試し、性能を評価した。ここでは、3 つのブログを対象に、いくつかの記事に対してどの程度正しく関連を抽出できるかまとめたものを示す。実験では、実際にシステムの判断が正しいか確かめるため、実験で抽出したすべてのトラックバックを確認した。今回取り上げたブログでは記事のキーワードがはっきりしていたので、そのキーワードについて少しでも触れていれば、関連ありと判断した。ブログ記事には関係なく、ブログサイト自体やブログ作者本人について書かれたものは関連なしと判断した。

表 1: トラックバック数が多い記事 (100 個以上)

		システムの判断	
		関連あり	関連なし
実際の結果	あり	60	12
	なし	26	134

表 1 は 1 つのブログ記事、表 2 は 2 つのブログ記事の結果をまとめたものである。表 1 の結果のうち、正しく判断できたのは約 83%、表 2 の結果のうち、正しく判断できたのは約 92%であった。

表 2: トラックバック数が少ない記事 (20 個以下)

		システムの判断	
		関連あり	関連なし
実際の結果	あり	5	2
	なし	0	18

システムで関係無しと判断したブログを確認したところ、表 1 に関しては、どれも一言触れているぐらいで主に別の話題について書かれたもの、つまり、もともとあまり関連が強くなかったものであった。また、表 2 に関しては、元記事より個人の感想が多く書かれていた。そのため、評価値が低くなってしまったと考えられる。

トラックバックスパムに関しては、ほとんど正しく関連無しと判断できていた。

5 考察

本研究で作ったシステムの特徴は以下のような点があげられる。

1. ブログの記事を指定するだけで、その記事に関連のあるブログ記事を抽出することができる。
2. Firefox の拡張として作っているため、余計なソフトを起動する必要などがない。
3. 解析できるブログが限られている。
4. 形態素解析時に MeCab が何度も起動するため邪魔である。

本システムでは、ブログ記事の URL を指定するだけで、その記事に関連のあるブログを抽出することができ、スパムトラックバックを見なくてすむので、トラックバックスパムを見たくない人や、普段トラックバックが使えなくて困っていた人にはとても便利になると考える。また元々トラックバックを使ったことがない人にとっても、本システムを使うことによって、新しいブログの探し方を得れるのではないかと考えている。

本システムは Firefox の拡張として作っているので、Firefox を使っている人なら誰でも使えるという点が大きな利点だと考えている。

また本システムでは、解析できるブログサービスを限定しているので、トラックバック数がたくさんあってもその全ての中から関連性を抽出しているわけではない。ブログサービスを限定した点に関しては、ひとつのブログサービスにはいくつかのテンプレートがあり、そのどこに本文が書いてあり、どこにトラックバックがあるかという情報を取得する必要があるが、種類が多すぎるブログなどにはまだ対応させていない。また本文を取得する際に文字化けしてしまうブログもあり、それらについては現時点では対応させていない。これらを改善するためには、多くのブログサービスのHTML を解析して、それらに対応するプログラムを書けばよい。

実装上の問題であるが、本システムでは形態素解析する回数だけ MeCab を呼び出すのだが、呼び出した時にウィンドウを開くのだが、これが解析するブログの数だけがでてしまう点が上げられる。トラック場草木の本文取得・単語の重要度計算および関連度の計算にかなり時間がかかっている。

6 おわりに

本研究では、トラックバックの中から記事の内容に関連のあるブログへのトラックバックのみを抽出するシステムを提案し、Firefox の Extension として実装した。これは、ブログ記事の URL を指定するだけで、トラックバックをたどり、元の記事とそのトラックバック先の記事を解析し、記事ごとに評価値を計算し、評価値が一定以上なら関連ありと判断し、ユーザに提示するシステムである。提案したシステムを使い、実際にどの程度関連記事を抽出できるのかさまざまなブログで試し、性能評価を行った。実験の結果、関連性のある記事を抽出することが出来ていることを確認した。特に、トラックバックスパムをフィルタリングすることができた。

今回提案したシステムでは、記事と記事の関連をどのように判断するかが難しく、多少同じような内容を書いていても、それが全体の一部だけだった場合は関連がないと判断しているため、ユーザーが見たかったブログ記事を見逃している可能性もある。記事と記事の関連を現在はきつく判断しているのため、ユーザが見たかった記事をフィルタリングしてしまっている可能性があり、今後の改良を検討している。

また対応させるブログを増やすことが重要だとも考えている。ブログサービスは 60 以上もあるといわれているので、今回対応していないブログサービスにも対応させていきたいと考えている。

参考文献

[1] 総務省　報道資料　ブログ及び SNS の登録者数　http://www.soumu.go.jp/s-news/2006/060413_2.html

[2] 日本ブログ協会 http://www.fmmc.or.jp/japan-blog/link/index.html

[3] blogWatcer http://blogwatcher.pi.titech.ac.jp/

[4] Blog Keyword Visualizer http://www.so-net.jp/web2/bkv/

[5] 風間一洋, 佐藤進也, 斉藤和巳, 木村昌弘：トラックバックネットワークに基づく SEO コミュニティの分析, 情報処理学会論文誌,2006

[6] 谷口智哉, 松尾豊, 石塚満：Blog コミュニティの抽出を分析, 第 6 回セマンティックウェブオントロジー研究会,2004

[7] Mozilla project http://www.mozilla.org/

[8] もじら組 http://www.mozilla.gr.jp/

[9] 松澤太郎, 下田洋志. Firefox の全て: C MAGAZINE, 10 月号特集, pp.36–71 (2005).

[10] XULPlanet http://www.xulplanet.com/

[11] MeCab http://mecab.sourceforge.jp/

連絡先

新美　礼彦
公立はこだて未来大学 システム情報科学部
　情報アーキテクチャ学科
〒 041–8655 北海道函館市亀田中野町 116–2
Phone:0138–34–6222　FAX:0138–34–6301
E-mail: niimi@fun.ac.jp