グラフ型データベースを用いたWikipediaからの関連概念抽出手法

大政 飛鳥† 井上 潮‡

†東京電機大学 工学研究科 情報通信工学専攻 〒120-8551 東京都足立区千住旭町5番
‡東京電機大学 工学部 情報通信工学科 〒120-8551 東京都足立区千住旭町5番
E-mail: †18kmc07@ms.dendai.ac.jp, ‡inoueu@mail.dendai.ac.jp

あらまし Wikipedia は誰でも自由に記事の編集や閲覧が行えるインターネット百科事典である．記事の網羅性の高さと量の膨大さから自然言語処理研究のコーパスとしても用いられてきた．本研究では記事が扱う概念同士の関連度の算出を目的とする．関連概念とはある概念から連想できる他の概念で，その情報は自然言語処理タスクに広く応用できる．本研究では記事同士を繋ぐリンクに着目した．他の記事へのリンクはリンク先記事との関連の存在を示すものと見なせるためである．しかしリンクを辿る処理は一般に計算コストが高い．このためデータの繋がりを辿る処理に優れることに加え，クエリの変更により異なる手法・タスクへの転用やデータの更新も容易なグラフ型データベース上で記事間リンクを解析し概念間の関連を抽出する手法を提案する．

キーワード Semantic relatedness, Web 構造マイニング，グラフ型データベース, Wikipedia

1. はじめに

Wikipedia は誰でも自由に記事の編集や閲覧が行えるインターネット百科事典である．2001 年に設立された日本語版の項目数は 2018 年 3 月には 110 万を超えた．扱う記事の内容は一般の百科事典では収録されていないような分野まで幅広く含む．これほど網羅性の高い言語リソースは他になく，膨大な数の記事中に存在する大量のテキスト情報や半構造化データ，カテゴリ構造や記事間のリンク構造などを持つ Wikipedia を情報源とした自然言語処理分野や人工知能，情報検索分野の研究がこれまで盛んに行われてきた[1].

1 つの記事が 1 つの概念に対応している Wikipedia において本研究では記事と記事の関連の抽出，つまり概念と概念の関連度の算出及び，ある概念に関連する他の概念の抽出を目的とする．この関連は is-a 関係や part-of 関係などを内包し区別はしない．概念同士の関連は Semantic relatedness ともいい情報検索や文書要約，単語の曖昧性解消といった自然言語処理タスクに基づく様々なアプリケーションで用いることができる[2].

本研究ではこの関連概念を抽出するために Wikipedia の記事中に存在する他の記事へのハイパーリンク（リンク）に着目した. Wikipedia のガイドラインでは記事の中で内容に関連する他の記事へのリンクを作成することが推奨されている[3]. 具体的にどの記事へリンクを作成するのかは編集者が決めており他の記事へのリンクはリンク先記事との関連の存在を明示するものと考えられる．

提案手法では関連概念を抽出したい記事（クエリ記事と呼ぶ）が関係するリンク，すなわちクエリ記事が含むリンクや逆にクエリ記事へ向かうリンクなどを辿ってまずは関連している可能性のある記事の候補となる集合を得る．その後クエリ記事と関連候補の記事との関連度をそれぞれ計算しその値が高いものを関連概念として抽出する．しかしデータの繋がりを辿っていく処理は一般に計算コストは高いという問題がある．そこで記事間リンクの解析にグラフ型データベースを用いた．グラフ型データベースとはグラフ理論に基づいたデータベースのことで，データそのものだけでなくデータ間の関係性も一緒に管理する．予めデータ同士の関係性が定義されているため効率的にデータにアクセスすることが可能である．さらにデータの操作はもちろんクエリの変更により異なる手法・タスクへの転用やデータの更新も容易である．

本論文の構成は以下のとおりである．2 章では Wikipedia が持つ研究リソースとしての特徴についてまとめ，3 章では概念間の関連度の算出を行う関連研究について述べる．4 章では本研究が用いるグラフ型データベースについて説明し，5 章でそのグラフ型データベースの構築方法について記述する．6 章で関連概念の抽出手法について述べ，7 章で手法の評価を実験によって行い，8 章でまとめを行う．

2. Wikipedia の特徴

Wikipedia は知識抽出のためのコーパスとしてこれまでに様々な研究で利用されてきた．研究リソースとして Wikipedia が持つ主な特徴を以下にまとめる．

- 記事の網羅性
- 記事の頻繁な更新
- 半構造化データ
- カテゴリ構造
- 密な記事間リンク
- URL に基づく概念の一意性
- 多言語サポート

まず記事の網羅性およびその頻繁な更新についてで

あるが, Wikipedia が扱う記事は一般的な概念に加えスポーツ選手やアーティスト, 企業, 映画作品やテレビ番組, 駅, 学術用語などと幅広い. これはインターネットを通じて誰でも自由に記事の編集や閲覧が行える Wikipedia ならではの特徴であり新しい概念の追加や最新情報の反映など編集活動が盛んに行われている. 記事には Infobox と呼ばれる半構造化されたデータが存在するものがあり, そこから知識を抽出することができる[4]. また編集者は内容に応じたカテゴリを記事に割り当てることができ, このカテゴリの構造や本研究で特に着目する密なリンク構造を利用して概念同士の関連や is-a 関係, part-of 関係を含んだ概念同士の関係抽出などが行われている. 記事のページは概念ごとに別けられ, それぞれは URL で区別することができる. 最後に多言語サポートであるが, Wikipedia は約 300 言語でサービスが提供されている. それぞれの記事には利用できる他の言語のページへのリンクが張られており, この情報は対訳抽出に利用できる[5].

3. 関連研究

関連概念を扱った分野でよく使われている手法に Gabrilovichr ら[6]の Explicit Semantic Analysis（ESA）がある. ESA とは Wikipedia の 1 つの記事が 1 つの概念に対応することを利用して単語を各記事（概念）に対するその単語の tf-idf 値に基づく要素とするベクトルで表現するものである. ベクトル同士のコサイン類似度を関連度として用いることで概念間の関連度を得ることが出来る. しかし記事のテキストを解析対象とする場合, 多義語の問題やそれが日本語で書かれていた場合の形態素解析が精度に影響を与える可能性がある.

一方, カテゴリ構造や記事間リンクを解析対象とする利点として前述した問題が起こりづらいことや手法が言語に依存しない点が挙げられる. ESA と同様 tf-idf をベースに, 記事間リンクの解析を行った研究として Milne[7] が提案する Wikipedia Link Vector Model（WLVM）や記事と記事間リンクを有向グラフと捉えて解析を行った Nakayama ら[8]の pfibf がある. また tf-idf 同様, 自然言語処理分野でよく使われる共起性解析を記事間リンクの解析に適用した伊藤らの研究[9]ではリンクの共起性が概念の関連度を表す指標として有効であることを示した.

4. グラフ型データベース

本研究では Wikipedia の記事間のリンクの解析にグラフ型データベースの Neo4j[10]を用いる. グラフ型データベースとはグラフ理論に基づいたデータベースで, データをノード, ノード同士の関係性を表すリレーションシップ（エッジ）, ノードやリレーションが持つ情報のプロパティの 3 つで管理する. またノードおよびリレーションシップの種類をラベルを用いて区別する. リレーショナルデータベースでは複数のテーブルを特定の列の値により動的に関係付けることによって柔軟にデータを取り出すことができるが, そのテーブルの結合処理はコストが高い. 一方グラフ型データベースの場合はデータそのものだけでなくデータ間の関係性も予め静的に格納しているので, リレーションシップで繋がれたデータに効率的にアクセスできる. Neo4j はオープンソースのグラフ型データベースで, データはグラフ構造に適した形式で保存される. そのデータには SQL に似た Cypher Query Language を用いて簡単に操作することができ, ページランクや最短経路探索, 中心性分析などグラフ理論に関するアルゴリズムも利用することができる.

5. データベースの構築

Wikipedia の記事をノード, リンクをリレーションシップとして Neo4j にインポートし関連概念の抽出を行うために, まず自作パーサを用いて Wikipedia が公開している日本語版の全ページが含まれた XML ファイル（pages-articles.xml, 18 年 5 月 24 日取得）から通常の百科事典の記事とそこに含まれるリンクを抽出した.

抽出された記事の中には「コンピュータ」と「コンピューター」のような表記ゆれや同義語などを 1 つの記事にまとめる目的で作成されたリダイレクト記事が存在する. 先の例で言うと「コンピューター」がリダイレクト記事であり, このページにアクセスすると「コンピュータ」のページに転送される仕組みになっている. 通常の記事とリダイレクト記事にはそれぞれ異なるラベルを付与して区別できるようにした. それと同時に抽出されたリンクの参照先がリダイレクト記事だったものは参照先をリダイレクト先の通常の記事へと書き換えている. これは記事間リンクを解析するにあたってリダイレクト記事が存在する概念の場合, 1 つの概念に対するリンク先の記事が複数に分散してしまうのを防ぐためである.

また, リンクのプロパティにはアンカーテキストおよび記事内におけるアンカーテキストの出現頻度を格納する. アンカーテキストからそのリンクが参照する記事を調べれば多義語の抽出や入力された語と記事の対応づけが, 記事からリンクを辿りそのアンカーテキストを調べれば記事の見出し語に対する同義語を抽出することが出来る.

そして記事内におけるアンカーテキストの出現頻度についてであるが, Wikipedia では 1 度記事の中でリ

ンクを作成した語が再び本文中に出てきたとしても全ての出現箇所ではリンクを作成しないよう推奨されている[3]．しかしある記事の見出し語が別の記事の本文中で何度も言及されていた場合，その 2 つの記事同士の関連は強いと考えることができる．よってアンカーテキストを記事の本文から最長一致で検索しマッチした回数を同じように記事中の全てリンクに対して求めた回数の和で除算した値を関連概念の抽出精度の向上に用いる．

以上の手順で作成したデータを実際に Neo4j へインポートした．具体的には通常の記事 1,107,203 件，リダイレクト記事 6,76,744 件，リンク 46,888,788 件で図 1 はインポートしたデータの一部を Neo4j で表示している様子である．

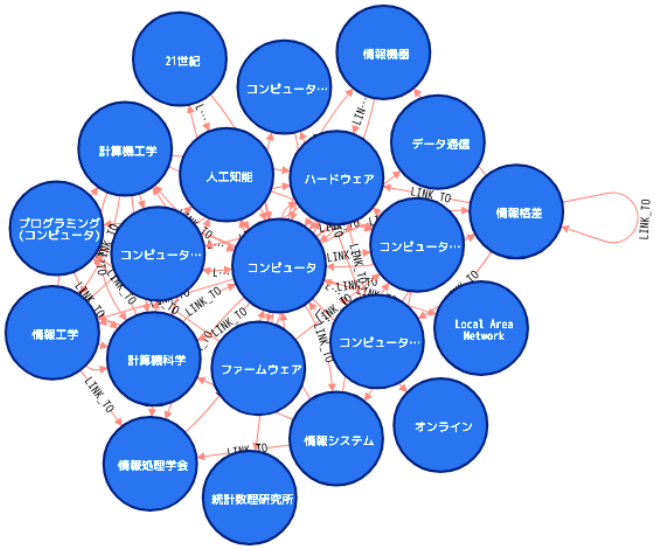

図 1. 記事「コンピュータ」とその記事がリンクしている記事(20 件)

6. 関連概念の抽出

提案するグラフ型データベース上で関連概念の抽出を行う手法は次の 2 段階で行う．(1) 関連候補概念（記事）の決定 (2) 関連度の算出．

6.1. 関連候補概念の決定

関連概念を抽出したい概念について書かれた記事をクエリ記事とする．クエリ記事と他の記事との関連度の計算を行う前に，その候補となる記事（候補記事）を選ぶ．これは関連が薄いと思われる記事まで関連度の計算を行い計算量が増えてしまうことを防ぐためである．

本論文では候補記事（概念）をクエリ記事からのホップ数が 1 である記事とした．つまりクエリ記事と直接繋がっている記事で，クエリ記事が本文中で参照している記事および反対にクエリ記事を参照している記事が含まれる．次節で詳しく述べる関連度計算ではクエリ記事に対する候補記事の関連度を，クエリ記事と候補記事の両方へリンクを作成している記事の数を使うためこのように決めた．

6.2. 関連度の算出

関連度を測る指標としてまずリンクの共起回数とアンカーテキストの出現頻度を用いる．リンクの共起とは異なる 2 つの記事へのリンクが同じ 1 つの記事の中で出現することとする．リンクが共起している回数が多ければ多いほど参照されている 2 つの記事の関連度は高いということになる．この共起の考え方を論文で例えると異なる 2 つの論文が同じ 1 つの論文から引用・または参考文献として示されていた場合，2 つの論文は共起しているといいその数が多ければ多いほど 2 つの論文は同じ分野やテーマを扱っている可能性が高いと判断するものである．このような関係にある記事を Neo4j では次のクエリで簡単に取得することができる．

```
MATCH
(a:Article)<-[:LINK_TO]-(b:Article)-[:LINK_TO]->(c:Article)
WHERE a.title = “コンピュータ” AND c.title = “人工知能”
RETURN b
```

このクエリは 2 つの記事「コンピュータ」と「人工知能」の両方にリンクを作成している記事を返す．このリンクの共起回数に基づく $cf(c_i, c_j)$ を以下のように決める．

$$cf(c_i, c_j) = \frac{df(c_i, c_j)}{df(c_i)} \quad (1)$$

ここで，$df(c_i, c_j)$ は記事 c_i と c_j の両方へリンクを作成している記事数であり $df(c_i)$ は記事 c_i へのリンクを作成している記事数とする．

一方アンカーテキストの出現頻度は 5 章で述べたように記事中におけるアンカーテキストの出現頻度でありクエリ記事と候補記事を結ぶ全てのリンクが持つ出現頻度の和を用いる．この値を $lf(c_i, c_j)$ として以下のように決める．

$$lf(c_j, c_i) = tf(c_i, c_j) + tf(c_j, c_i) \quad (2)$$

ここで，$tf(c_i, c_j)$ は記事 c_i の本文中の記事 c_j へのリンクが持つアンカーテキストの出現頻度とする．

しかしこの 2 つの指標のみで関連度の計算を行うと「日本」や「2016 年」といった記事の内容にかかわらず多くの記事からリンクが作成されている概念の場合，共起回数が多くなり関連度が不当に高くなってしまう．そのため関連度計算を行う記事へのリンク数，つまり被リンク数を見てその数が多ければ多いほど関連度を

下げるという対策をとった．具体的には関連研究と同様に tf-idf の考えを使い被リンク数の多さに応じて関連度を下げた．

$$idf(c_j) = log\frac{N}{df(c_j)} \quad (3)$$

ここで，Nは全記事数（リダイレクト記事を除く），$f(c_j)$は記事c_jへのリンクを作成している記事数とする．

以上からクエリ記事を記事c_i，候補記事をc_jとして記事c_iに対するc_jの関連度$sr(c_i, c_j)$を次のように定義する．

$$sr(c_i, c_j) = cf(c_i, c_j) \cdot idf(c_j) \cdot \left(1 + lf(c_i, c_j)\right) \quad (4)$$

Neo4j に対して候補記事の抽出および関連度の算出を行うクエリを発行し，関連度が高いものを関連概念として抽出する．

7. 実験と考察

7.1. 実験概要

提案手法を評価するために 3 つの実験を行った。まず 5 章で構築したデータベースに対して 6 章で述べた手法に基づくクエリを実行し関連概念が抽出できることを確認した．次に関連概念の抽出にかかる時間を調べるために無作為に選んだクエリ記事に対して関連概念が 5 件以上抽出できたものが 1 万記事になるまで繰り返し行った．最後にグラフ型データベースを用いる本手法の優位性を示すためにデータベースはそのままにクエリを変更することで他のタスクにも利用できることを確認した．

7.2. 実験環境

実験は表 1 に示す環境で行った．

表 1. 実験環境

項目	値
OS	CentOS 7.6
CPU	Core i7-3770
メモリ	24GB
ストレージ	SSD 128GB
データベース	Neo4j Community Edition 3.5.2

7.3. 結果と考察

図 2 のように Neo4j 上でクエリを実行し，関連度が高いものから 10 件を関連概念として抽出した結果を表 2 に示す．

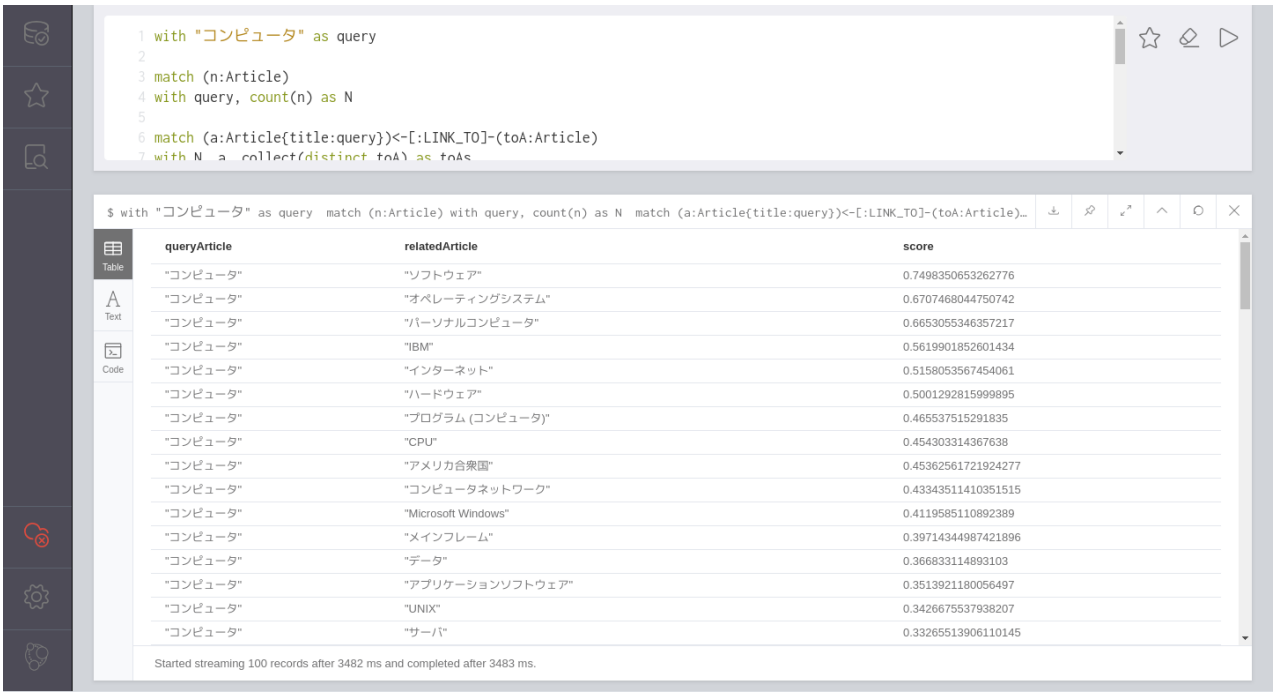

図 2. クエリを実行した様子

表 2. 抽出された関連概念

クエリ記事	抽出された関連概念（上位 10 件）
トランペット	トロンボーン，ホルン，ティンパニ，チューバ，ジャズ，オーケストラ，金管楽器，フリューゲルホルン，コルネット，アメリカ合衆国
HTML	CSS，XML，ウェブブラウザ，Javascript，XHTML，W3C，ウェブページ，Java，WWW，PDF
山手線	東日本旅客鉄道，京浜東北線，東京都，東京メトロ，東京駅，新宿駅，埼京線，品川駅，湘南新宿ライン，渋谷駅
冬	夏，春，秋，雪，北海道，日本，気温，川，九州，2008 年
グラフ理論	数学，連結グラフ，2 部グラフ，アルゴリズム，閉路，木(数学)，次数(グラフ理論)，隣接行列，グラフ彩色，計算機科学
ネコ	イヌ，ネズミ，ウサギ，日本，ペット，ウマ，動物，鳥類，ヒト，キツネ
非接触型決済	QUICPay，電子マネー，クレジットカード，FeliCa，JCB，おサイフケータイ，SmartPlus，PASMO，デビットカード，楽天 Edy
登山	山，写真，スキー，ハイキング，登山道，キャンプ，富士山，登山家，山小屋，飛騨山脈
日経平均株価	東京証券取引所，東証株価指数，ダウ平均株価，株式，株価，2008 年，円相場，日本経済新聞，リーマンショック，トヨタ自動車
牛丼	吉野家，すき家，すき焼き，外食産業，カレーライス，日本料理，松屋フーズ，牛肉，チェーンストア，2008 年

次に関連概念の抽出にかかる時間については Wikipedia 日本語版(記事数:1,107,203，リンク数: 46,403,446)の場合，クエリ記事 1 件あたりの平均は **1.47 秒**となった. Neo4j は 1 つのクエリをシングルスレッドで処理しており，実験環境に合わせて最大 8 件のクエリを同時に実行して行ったこの実験全体にかかった時間は 30 分であった．ただし平均計算時間は述べたように 1.47 秒だったが，ごく一部の記事では 30 秒を超えた記事も存在した．図 3 は計算時間を降順にならべた時の順位と対応する計算時間である．

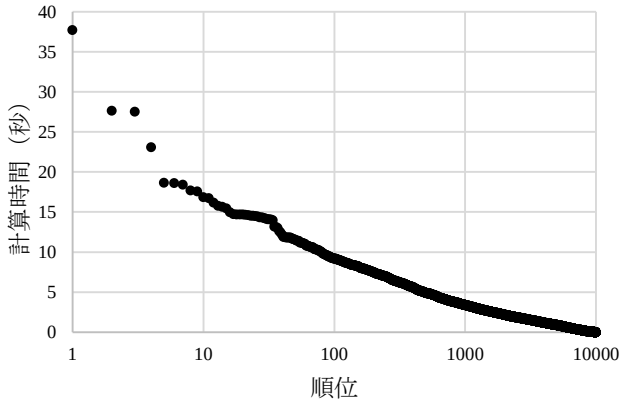

図 3. 計算時間の分布

今回の実験で特に時間がかかった上位 3 件のクエリ記事とその時間はそれぞれ「8 月 1 日」(37.7 秒)，「10 月 7 日」(27.6 秒)，「8 月 12 日」(27.5 秒)であり，どれも多くの記事からリンクが作成されている記事であった．図 3 のような分布になった理由としては本手法が必要とする計算量が候補記事の数およびリンクの共起を調べる必要のある記事数（候補記事それぞれが持つ被リンク数）に依存することに加え，図 4 のようにその被リンク数の分布がごく一部の記事に集中している（Zipf 分布）ためと考えられる．

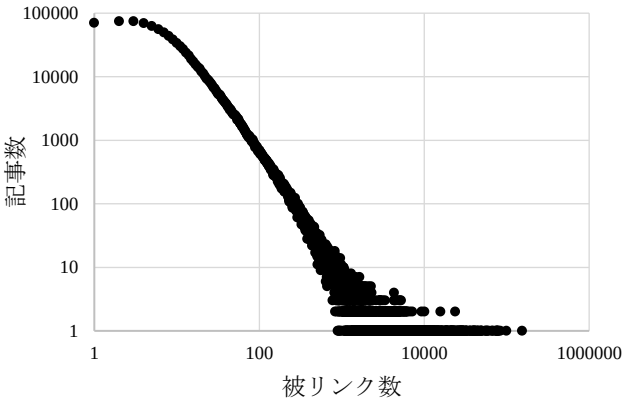

図 4. 被リンク数の分布

リンクが集中している記事の例としては前述した日付を扱った記事の他に「日本」や「2008 年」などリンク元記事の内容にかかわらず多く記事から広くリンクが作成されている記事であり，そのような記事をクエリ記事にしたり候補記事の中に多く含まれていたりすると計算量は増加してしまう．多くの場合「2008 年」といった年は関連概念として適当ではないと考えられ，候補記事のタイトルが「年」で終っている記事を除き同様の実験を行った場合のクエリ記事 1 件あたりの平均計算時間は **0.815 秒**であった．本手法の計算量がクエリ記事まわりの局所的な状況に左右されることを確かめるために，使用するデータを Wikipedia 英語版(記事数:5,690,941，リンク数: 138,930,334)に替えても行ったが，1 件あたり **1.07 秒**と全体の記事数およびリンク数は増えているにも関わらず日本語版の場合と比べて計算時間が短くなることを確認した．日本語版の記事 1 つに対するリンク数は単純計算で約 41.9 であるのに対し英語版では約 24.4 と英語版の方が記事間リンクは疎であり，その分計算時間が短くなったと考えられる．これは全体のデータ量にかかわらず局所的な探索が可能であるグラフ型データベースの特徴でもある．

最後にクエリの変更による他のタスクへの利用であるが，5 章で述べたようにアンカーテキストの情報

から同義語を抽出することが出来た．具体的には同義語を抽出したい記事へのリンクを取得し，そのアンカーテキストを同義語として抽出した．表 3 は各記事へのリンクが持つアンカーテキストを使われた回数が多かった順に同義語として 3 件並べている．

表 3. アンカーテキストから抽出された同義語

記事の見出し語	同語義(上位 3 件)
コンピュータ	コンピューター，電子計算機，デジタルコンピュータ
DVD	DVD-ROM, DVD-RW，ライブ DVD
住宅	民家，住居，邸宅
エレベーター	エレベータ，昇降機，リフト
寿司	すし，押し寿司，鮨

また Neo4j の場合，ノードやリレーションシップが持プロパティに対して全文検索を行うことが出来る．これを利用して語「JR」でアンカーテキストを検索し，そのリンクの参照先を辿ることによって語「JR」が指す記事（概念）の抽出を行った．参照先として多かった順に「東日本旅客鉄道」，「西日本旅客鉄道」，「東海旅客鉄道」，「九州旅客鉄道」，「四国旅客鉄道」である．他にも語「グラフ」に対しては記事「グラフ(関数)」，「グラフ理論」，「統計図表」などが，語「メタル」では記事「ヘヴィメタル」などが得られた．これは関連概念の抽出にあたって入力語からクエリ記事への対応づけを行う際にも利用することが出来る．他にもグラフ理論に関するアルゴリズムを利用した解析やドライバを経由したアプリケーションとの連携のしやすさから Wikipedia 解析のプラットフォームとしての発展が期待できる．

8. まとめと課題

本論文ではグラフ型データベースを用いた Wikipedia からの関連概念の抽出手法について述べた．実際に関連概念の抽出を行えることを確認し日々記事の数が増える Wikipedia の解析にも対応できることを示した．1 件あたりの計算時間は候補記事の選び方を工夫することで改善できる可能性がある．例えばデータベースにカテゴリのデータを追加し，その関係性から候補記事を絞るといった方法が考えられる．抽出の精度に関してはこの種のタスクの評価によく用いられているデータセット WordSimilarity-353 Test Collection[11,12]による評価を検討していたが，データセットに含まれる語句の中には多義語を含み Wikipedia の記事との対応づけが必要である点や，その語句には一般的な語が多く既存の連想シソーラス辞書などでは扱っていない固有名詞を多く含む Wikipedia をソースとした概念間の関連度の評価に用いるデータセットとしては問題が生じた．一方で関連概念が抽出できる概念（語）が Wikipedia 記事の見出し語に限定されることは本手法の欠点でもある．今後，複数の被験者による評価や新たな評価用データセットの作成などを検討する．

しかしながら Wikipedia からの知識抽出のプラットフォームとしてグラフ型データベースを利用することには発展の余地があると考える．

参 考 文 献

[1] 中山浩太郎，伊藤雅弘，M. Erdmann，白川真澄，道下智之，原隆浩，西尾章治郎，“Wikipedia マイニング：Wikipedia 研究のサーベイ”，情報処理学会論文誌データベース Vol2, No4, pp. 49-60, 2009.

[2] A. Budanitsky, “Lexical Semantic Relatedness and Its Application in Natural Language Processing”, Ph.D. thesis, University of Toronto, Ontario, 1999.

[3] Wikipedia: 記事どうしをつなぐ，https://ja.wikipedia.org/wiki/Wikipedia:記事どうしをつなぐ

[4] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, “DBpedia: A Nucleus for a Web of Open Data”, Proc. of International Semantic Web Conference, Asian Semantic Web Conference, pp. 722-735, 2007.

[5] 胡寅駿，林良彦，永田昌明，“Wikipedia Infobox の言語間対応付けによる対訳辞書の抽出”，言語処理学会 第 19 回年次大会 発表論文集，pp. 398-401, 2013.

[6] E. Gabrilovich. and S. Markovitch, “Computing semantic relatedness using Wikipedia-based Explicit Semantic Analysis”, Proc. of the International Joint Conference on Artificial Intelligence, pp. 1606-1611, 2007.

[7] D. Milne, “Computing semantic relatedness using wikipedia link structure”, Proc. of the New Zealand Computer Science Research Student Conference, pp. 1-8, 2007.

[8] K. Nakayama, T. Hara and S. Nishio, “Wikipedia Mining for An Association Web Thesaurus Construction”, Proc. of International Conference on Web Information Systems Engineering, pp. 322-334, 2007.

[9] M. Ito, K. Nakayama, T. Hara and S. Nishio, “, Association thesaurus construction methods based on link co-occurrence analysis for Wikipedia”, In 17th ACM Conference on Information and Knowledge Management, pp. 817-826, 2008.

[10] Neo4j, https://neo4j.com/

[11] L. Finkelstein, E. Gabrilovich, Y. Matias, E. Rivlin, Z. Solan, G. Wolfman, and E. Ruppin, “Placing Search in Context: The Concept Revisited”, ACM Transactions on Information Systems, 20(1):pp. 116-131, 2002

[12] The WordSimilarity-353 Test Collection, http://www.cs.technion.ac.il/~gabr/resources/data/wordsim353/