人工知能学会研究会資料
SIG-SWO-037-04

日本語版WC3(Wikipedia Category Consistency Checker) - 日本語版Wikipediaのカテゴリに所属するページのメタデータの一貫性の分析 -

Japanese version WC3(Wikipedia Category Consistency Checker) - Analysis of the consitency of metadata that belongs to same Wikipedia categories -

吉岡真治[1*]
Masaharu Yoshioka[1]

[1] 北海道大学大学院情報科学研究科
[1] Graduate School of Information Sciecne and Technology, Hokkaido University

Abstract: We have already proposed WC3 (Wikipedia Category Consistency Checker) that supports to evaluate consistency of the category information in Wikipedia by using DBPedia information. In this paper, we propose a Japanese version WC3 that analyzes Japanese version Wikipedia by using Japanese Wikipedia. We discuss the problem of the English version WC3 and difference between the ammount of the metadata in English DBPedia and Japanese one. Based on the discussion, we propose a new algorithm to construct appropriate SPARQL queries for the Wikipedia categories. We also discuss the analysis result of the system.

1 はじめに

DBPedia[1] は、Web 上の百科事典である Wikipedia[1] のページから抽出されたメタデータに基づいて構築された大規模な事象に関する構造化情報のデータベースであり、Linked Open Data[2] の中心として、様々なデータと関連づけられて利用されている。この DBPedia の情報は、Wikipedia 中に記述された構造化情報 (主に、Infobox に記述) から抽出されるため、その情報の品質は、Wikipedia におけるこれらのメタデータの記述の一貫性に依存することになる。

この様な状況を分析するために、我々は、WC3(WC-triple:Wikipedia Category Consistency Checker) を提案している [3]。本システムは、これまでの Wikipedia の品質についての議論の際に行われてきた内容的な分析 [4, 5] や、ページの編集に携わった人々に関する属性を用いた分析 [6] ではなく、Wikipedia 中の構造化情報の記述スタイルの一貫性 (特に、同一のカテゴリに属するページ群における一貫性) をチェックする。具体的には、“Songs written by Paul McCartney” のように、クラス (song) とトピック (Paul McCartney) の組合わせで表されるようなカテゴリに対して、DBPedia のメタデータデータベースを参照しながら、そのカテゴリに属するページをできるだけ過不足無く抽出可能な SPARQL クエリの作成を行い、その検索結果と実際のページ群を比較することにより、メタデータの不足、新しくカテゴリに追加するべきページの発見などを支援する。

本研究では、この WC3 において、データベースとして、日本語 Wikipedia とそこから抽出した情報に基づく日本語 DBPedia を利用することによって、日本語版 WC3ja を提案する。また、従来の WC3 における問題点について議論し、その問題を解決するための改良版アルゴリズムについても提案を行う。さらに、その分析結果を踏まえ、日本語版 DBPedia と英語版 DBPedia の現状についての考察を行う。

*連絡先： 北海道大学大学院情報科学研究科
〒 064-0806 札幌市北区北 14 条西 9 丁目
E-mail: yoshioka@ist.hokudai.ac.jp

[1] http://en.wikipedia.org/

2 WC3(WC-triple:Wikipedia Category Consistency Checker)

2.1 システムの動作アルゴリズム

Wikipedia のカテゴリには、「日本」、「ポール・マッカートニー」といったトピックを表すようなカテゴリ、「作家」、「歌」などのクラスを表すようなカテゴリ、「日本の作家」などのトピックとクラスの組み合わせによりあらわされるカテゴリが存在する。WC3(WC-triple:Wikipedia Category Consistency Checker) は、この中のトピックとクラスの組み合わせにより作られたカテゴリーを対象に、DBPedia の情報を用いて、そのカテゴリーの情報を表す適切な SPARQL クエリを作成し、その検索結果とカテゴリに属するページ群を比較することによって、ページ群に付与されているメタデータの一貫性について分析を支援するシステムである。

本システムは、与えられた Wikipedia カテゴリに対して、以下のようなアルゴリズムによって、SPARQL クエリを作成すると共に、その検索結果の表示を行っていた [3]。

1. カテゴリを入力とし、そのカテゴリに属するページ集合から他のページへのリダイレクトとなっているページを除いた集合 Pc を抽出する。

2. 抽出した全てのページがもつ異なり属性からカテゴリに関する属性[2]を除いた全ての異なり属性について、各属性 $(a_1, ..a_n)$ が存在するページの集合 $PP_1, ..PP_n$、全データベース中でその属性を持つページの集合 $PA_1, ..PA_n$ を用いて、精度 $p_i = |PP_i|/|PA_i|$、再現率 $r_i = |PP_i|/|Pc|$、F 値 (精度と再現率の調和平均) を計算する。

3. F 値の最も高い属性をクエリの候補とするとともに、上位 10 件を組み合わせのために用いる属性の候補とする。

4. 3 で求めた属性では精度と再現率のバランスを考慮するために、クラスを表すような一般的な属性が候補に含まれない可能性がある。そのため、網羅性を考慮した属性の候補を、各親カテゴリについて次のような手順で作成し、組合わせ属性の候補として追加した。

   (a) 各親カテゴリについて、共通の親を持つ兄弟カテゴリ (例えば、「Songs written by Paul McCartney」の親カテゴリ「Songs by songwriter」に関する兄弟カテゴリ「Songs written by Bob Dylan」など) を 5 つランダムに抽出し (兄弟カテゴリが 5 以下の場合は全てを利用)、ページの集合を作成する。

   (b) 2 の手順と同様に、このページ集合が持つすべての異なり属性について、精度、再現率、F 値を計算する。この時、網羅性を考慮して、再現率が 0.9 以上 (間違いや、個物を表さないページなどがある場合を考慮して、多少の検出漏れは許容する) の属性のうちで、F 値の高いもの 2 件を組み合わせの候補として追加する。

5. 候補となった属性を 2 つ組み合わせたクエリを作成し、同様に、精度、再現率、F 値を計算し、F 値の最も高いものをクエリの候補とする。ただし、RDF のトリプルで表されている属性のうち、対象を共有するものについては、主に、同一のトピックに関する属性の組み合わせになるため、組み合わせの候補から除外している。

このようにして作成したクエリを満たすページの集合と、対応するカテゴリのページについて、比較することにより、以下の 3 種類のページの情報を収集する。

Found クエリにより見つけられたカテゴリのページ

NotFound クエリにより見つけられなかったカテゴリのページ
二つの属性の組み合わせのクエリの場合には、不足している属性の情報を合わせて示す。

Error クエリにより見つけられたがカテゴリに属さないページ
このようなエラーページを排除するためのクエリを作成するための情報として、複数の Error ページに共通し、カテゴリに属するページには、ほとんど含まれない属性の情報を示す。

2.2 システムに関する問題点

本システムは、“Songs written by 〜” や “Films directed by 〜” のようなトピックが人名で表されるような場合には、多くの場合、一般的な SPARQL クエリが作成可能であったが、 “People from London” などの様に、トピックの表記方法にバリエーションがあるような場合 (birthPlace に対応するメタデータとして「London」、「London, England, UK」、「Lewisham, London, England」などが存在) にうまくクエリを作ることができなかった。

[2]カテゴリに関する参照を行っている属性と、カテゴリを情報源として作成されている Yago の情報については、候補から除外している。

また、本システムでは、全てのカテゴリに属するページについてメタデータを収集し分析していたため、多くのページが属するカテゴリを解析する際に、長時間のデータがかかるという問題があった。さらに、候補となる異なり属性の数が増えた場合に、全データベース中の出現回数の計算が必要になる precision の計算にも時間がかかるという問題もあった。

3 日本語版 WC3(WC3ja)

3.1 プロトタイプシステムによる予備実験

前節で提案した WC3 について、メタデータとして日本語版 DBPedia[3]、解析対象として日本語版 Wikipedia を用いることにより、日本語版 WC3 のプロトタイプを作成し、その評価を行ったところ、英語版の WC3 においてうまくクエリが作成できたカテゴリの日本語版 (「〜の製作した楽曲」や「〜の監督作品」など) についても、適切な SPARQL クエリが作成できない場合が多く存在した。

一番大きな問題は、クラスを表すようなメタデータを発見するのに失敗しているという問題であったため、日本語版 DBPedia と英語版 DBPedia[4]におけるクラス情報の付与数についての比較を行った (表 1)。

表 1: クラス定義の数の日英比較

クラス定義の数	日本語版	英語版
0	582,360	2,339,705
1	20,536	225,672
2	14,291	891,513
3	4,984	105,579
4	20,503	192,683
5	43,970	298,393
6-	238,233	3,138,894
総計	924,877	7,192,439

この結果を、日本語の DBPedia では、63% (582,360/924,877) のページにクラス情報が存在しない (英語版では、33% (2,339,705/7,192,439)) という事が確認された。この結果を踏まえると、前節で提案したアルゴリズムのように、兄弟カテゴリとあわせて、再現率が 0.8 以上といった制約を利用した場合に、ほとんどの場合に、クラスに関する情報を見つけられないという状況になったと考えられる。

[3]2014/12/30 版のデータを利用

[4]2014 年版の DBPedia のデータを利用 http://wiki.dbpedia.org/Datasets2014/

3.2 改良アルゴリズムの提案

日本語版 WC3 におけるプロトタイプにおいては、英語版 WC3 と同じアルゴリズム、同じ閾値などを使った場合に、うまく動作しないことが確認された。この問題を解決するために、閾値を下げることも検討したが、兄弟カテゴリを使った場合に、そのカテゴリにどれくらいクラス情報が付与されているかに応じて結果が大きく異なると言った問題が発生した。

そこで、改良アルゴリズムでは、兄弟カテゴリの利用をせず、カテゴリに属するページのみに注目することとした。ただし、この場合には、トピックに関するテンプレートなどといったクラスとは異なる情報が、選択されることがあったため、クラスに関する情報を表す述語を持つメタデータ (述語が http://www.w3.org/1999/02/22-rdf-syntax-ns#type) に限定して、候補を選択することとした。

また、改良アルゴリズムでは、2.2 節で述べた問題に対応するため、以下の二つの方法により高速化を実現する。

1. カテゴリに属するページ数が閾値 pst (pageSizeThreshold) 以上の場合には、pst 件のサンプルを利用して分析を行う[5]。

2. SPARQL クエリの候補となる属性の recall が低い場合には、F 値が高くなることが期待できないため、カテゴリに関する異なり属性の情報を収集した際に、閾値 rt (recall threshold) よりも低い属性については、全データベース中の出現回数の計算が必要になる precision を計算しない。

これらの高速化に加え、FILTER 構文を利用することにより、より適切な SPARQL クエリを作成できるように修正を行う。ただし、FILTER 構文で用いる文字列の候補は、クラスとトピックを表すような文字列に限定することで、カテゴリ名に即した SPARQL クエリを作成するようにした。

この改良アルゴリズムの基本的な手順は、前節で述べた従来の WC3 のものと同じである。変更部分を明確にするため、変更部分には下線を引いて，従来部分と区別する。

1. カテゴリを入力とし、そのカテゴリに属するページ集合から他のページへのリダイレクトとなっているページを除いた集合 Pc を抽出する。

2. 抽出したページの数がpst を越える場合には、pst 件のページを抽出し、ページ数がpst 以下の場合は、全てのページを利用する。このページ集合をPt とする。

[5]SPARQL クエリの limit を用いて選択

(a) これらのもつ異なり属性からカテゴリに関する属性[6]を除いた全ての異なり属性について、各属性 $(a_1, ..a_n)$ が存在するページの集合 $PP_1, ..PP_m$ を計算する。この時、$|PP_i| \geq rt \times |Pt|$ を満す属性について、全データベース中でその属性を持つページの集合 $PA_1, ..PA_n$ を用いて、精度 $p_i = |PP_i|/|PA_i|$、再現率 $r_i = |PP_i|/|Pt|$、F 値 (精度と再現率の調和平均) を計算する。

(b) FILTER 構文で用いる文字列としては、トピックやクラスを表すキーワードに限定する。そのため、兄弟カテゴリとの文字列比較を行い、共通部分を除去することで、トピックやクラスを表すキーワードを作成する (例：「山下達郎の楽曲」について、兄弟カテゴリ「槇原敬之の楽曲」などを用いると、「山下達郎」を抽出し、「山下達郎のアルバム」を用いると、「楽曲」を抽出する)。上記の属性において、そのトリプルの目的語にこれらの文字列を含む場合は、上記の属性の目的語部分を変数化して、FILTER 構文と組み合わせたものを候補として追加する。これらの候補についても、F 値を計算する。

3. 属性、もしくは、目的語を変数化して FILTER 構文と組み合わせたもののうち、F 値の大きな方から上位 10 件を組合わせのために用いるクエリのようその候補とする。

4. 3 で求めた属性や FILTER 構文の結果 では精度と再現率のバランスを考慮するために、クラスを表すような一般的な属性が候補に含まれない可能性がある。そのため、網羅性を考慮した属性の候補を、各親カテゴリについて次のような手順で作成し、組合わせ属性の候補として追加した。

   (a) 利用するページのうち、述語として、http://www.w3.org/1999/02/22-rdf-syntax-ns#type を持つものに限定し、その再現率を計算する。再現率が trt(typerecallthreshold) 以上のものに限定し、F 値の高いもの 2 件を組み合わせの候補とする。

5. 3 で作成した候補集合と、クラスの候補となる属性を組み合わせて候補となる SPARQL クエリを作成する。クラスの候補となる属性が存在しない場合には、3 で作成した候補集合を単独で利用する。これらの候補について、同様に、精度、再現率、F 値を計算し、F 値の最も高いものをクエリの候補とする。

[6]Yago の情報については、同様に候補から除外している。

3.3 システムの実装

先に述べた改良アルゴリズムを用いて日本語版 WC3 (WC3ja) を作成した[7]。本システムでは、2014 年 12 月 30 日版の日本語版 DBPedia[8]の情報を用い、パラメータとしては、$pst = 50$,$rt = 0.2$, $trt = 0.6$ を利用した。

図 1 は、「槇原敬之が製作した楽曲」(リダイレクトを除くページ数：56 件) に対する結果であり、構築された SPARQL クエリは、以下に示すように、シングル曲で作者の情報に槇原敬之という文字列を含むというものとなった。

```
SELECT ?s
WHERE {?s http://dbpedia.org/ontology/writer ?o0 .
FILTER regex(?o0, "槇原敬之")
?s http://www.w3.org/1999/02/22-rdf-syntax-ns#type
http://dbpedia.org/ontology/Single .
MINUS { ?s
<http://dbpedia.org/ontology/wikiPageRedirects>
 ?o . }}
```

56 ページ中の 45 ページがこのクエリを満し (Found:45 ページ、NotFound:11 ページ) となった。クエリが FILTER 構文を利用したものになった原因としては、作詞・作曲などに対応する infobox において、”槇原敬之”に、Wikipedia ページへのリンクがついているものと、ついていないものが存在したためである。また、NotFound のページについては、infobox に作詞・作曲の情報が存在しない場合や、infobox 自体が存在しないために、Single 曲でも Single というクラスが与えられていないページがあった。

また、多くのページ数が属する例として、「東京都出身の人物」についても分析を行った。この結果、以下のクエリが作成された。100,000 ページ中の 2295 ページがこのクエリを満し (Found:2,295 ページ、NotFound:7705 ページ、Error:2,608 ページ) となった。

```
SELECT ?s
WHERE {?s http://dbpedia.org/ontology/birthPlace
?o0 . FILTER regex(?o0, "東京都")
?s http://www.w3.org/1999/02/22-rdf-syntax-ns#type
http://dbpedia.org/ontology/Person .
MINUS { ?s
<http://dbpedia.org/ontology/wikiPageRedirects>
 ?o . }}
```

多くの birthPlace の情報では、「東京都〜区」や、「東京都, 日本」といった表現も存在したが、FILTER 構文を使うことにより、この様な情報についても、一定レベルで分析を行うことができるようになったことを確認した。

NotFound のページを分析すると、infobox 中に出身地を表すページが多く含まれていることが確認された、さらに、DBPedia を分析すると、birthPlace ではなく、

[7]http://wnews.ist.hokudai.ac.jp/wc3ja/
[8]http://ja.dbpedia.org/

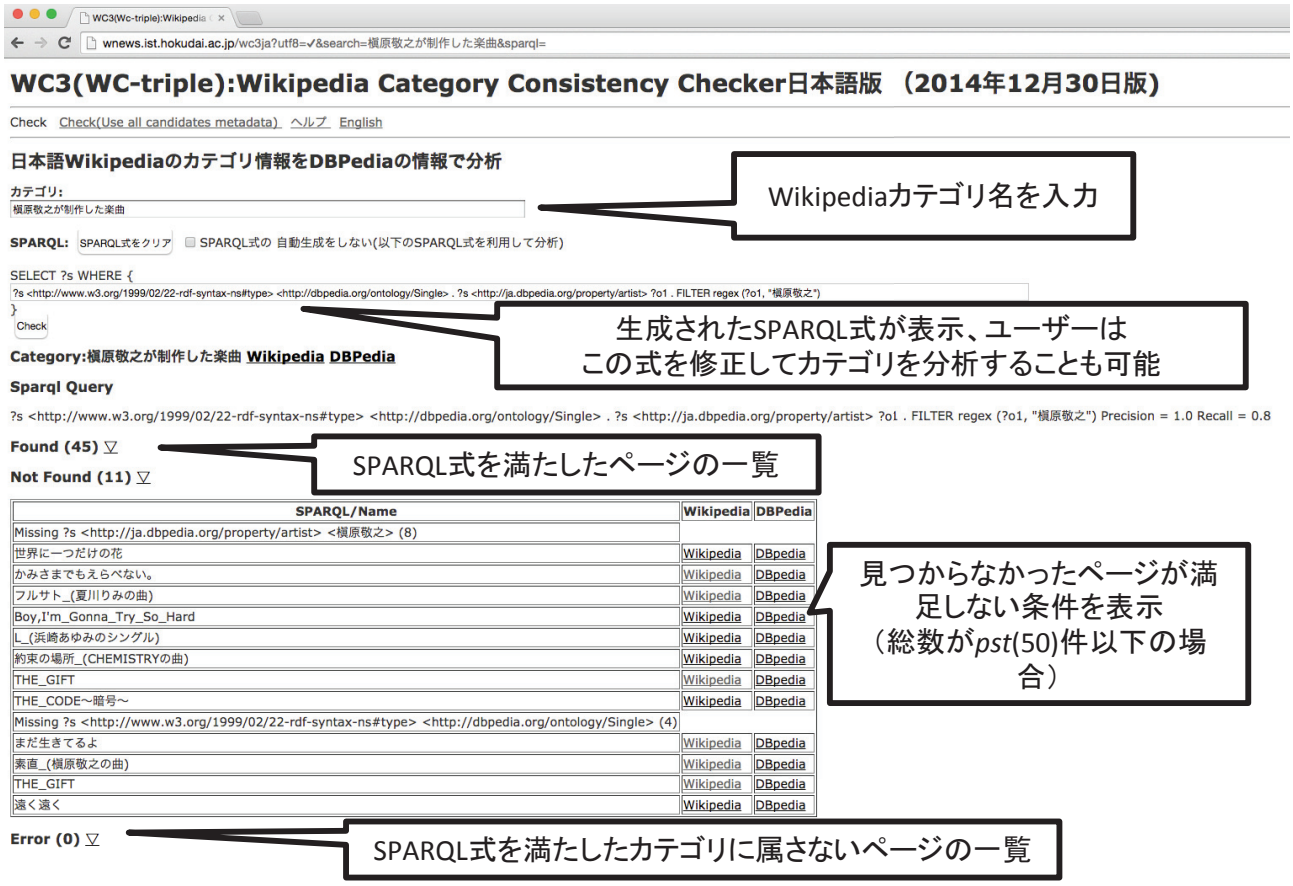

図 1: WC3ja(Wikipedia Category Consistency Checker) の実行例

hometown という関係で結ばれている事が多い事が分かった。そこで、上記のクエリを手作業で修正し、以下のようなクエリを作成した。

```
SELECT ?s
WHERE {?s http://dbpedia.org/ontology/hometown
?o0 . FILTER regex(?o0, "東京都")
?s http://www.w3.org/1999/02/22-rdf-syntax-ns#type
http://dbpedia.org/ontology/Person .
MINUS { ?s
<http://dbpedia.org/ontology/wikiPageRedirects>
 ?o . }}
```

その結果は、100,000 ページ中の 832 ページがこのクエリを満すことが確認された。このことは、同じ出身地という表現に対して、異なる関係として抽出されていることを示している。

同様のクエリを英語版 DBPedia(People from Tokyo：1,600 ページ) に与えたところ、以下の birthPlace のクエリでは、688 ページ、birthPlace を hometown に置き換えたクエリでは 32 ページといった明らかな量的な違いがあった。こちらを見ると、主に、Born に関する地名が birthPlace に Origin が hometown という違いがあるように思われる。日本語版では、この違いが infobox 上の表記で明確でない事が、この様な結果に結び付いた可能性は考えられる。

また、英語版の Wikipedia においても、birthPlace のクエリでは、Error が 1010 ページ見つかっており、(People from Tokyo：1,600 ページ) のような英語圏の人にとってはローカルな話題については、そのカテゴリの網羅性は、あまり高くないことも確認された。

```
SELECT ?s
WHERE {?s http://dbpedia.org/ontology/birthPlace
?o0 . FILTER regex(?o0, "Tokyo")
?s http://www.w3.org/1999/02/22-rdf-syntax-ns#type
http://dbpedia.org/ontology/Person .
MINUS { ?s
<http://dbpedia.org/ontology/wikiPageRedirects>
 ?o . }}
```

3.4 考察

本システムで提案した高速化の手法を用いることにより、100,000 件の情報を持つような「東京都出身の人物」といったクエリに対しても、10 秒以内で結果を返すことが出来るようになった。まだ、十分なスピード

とはいえないものの、1 分以上かかっていた以前のシステムと比較すると、実用性は向上したと考えている。

また、FILTER 構文に対応したことにより、リンクがついたりついていなかったりすることによる違いを無視した分析や、地名、年号などのようなトピックの際に、多く現れる RDF-triple の Object のバリエーションを考慮した SPARQL クエリが構築できるようになり、これまでよりも多くのカテゴリについて分析を行うことが可能になった。

例えば、「1980 年生」というカテゴリを分析すると、3,201 ページ中 1,005 ページが見つかり、Error として、23 ページが見つかった。この中の多くは、単純に、「〜年生」といった記述が存在しないだけであるが、「フレッド・ルイス」のように、「1983 年生」といった間違ったカテゴリが付与されている[9]場合なども見つけることができた。これは、カテゴリの排他性 (例えば、「〜年生」に関するカテゴリはひとつしか持てない) などと組み合わせることによって、カテゴリ情報と DBPedia が抽出の対象としている infobox の情報の食い違いなどを見つけることが可能になると考えられる。

4 おわりに

本論文では、英語版の Wikipedia と DBPedia の情報に基づいて、Wikipedia におけるカテゴリに属するページにおけるページが持つ構造化情報の記述スタイルの一貫性をチェックする WC3 の日本語化を、日本語版の Wikipedia と DBPedia の情報を用いて作成する日本語版 WC3(WC3ja) を提案した。

本システムでは、日本語版 DBPedia と英語版 DBPedia の違いを考慮したパラメータの変更を行うと共に、FILTER 構文を用いた柔軟な SPARQL クエリの構築を支援すると共に、カテゴリに多数のページが属するときに、システムのレスポンスが非常に悪くなる問題に対応するための改良を行った。この改良により、10 万件のページを持つカテゴリについても 10 秒程度で分析ができるとともに、FILTER 構文を用いて、従来の WC3 ではうまく SPARQL クエリが作れないようなカテゴリの分析も可能となった。

今後は、ツールを実際に使ってもらいながらフィードバックをもらうとともに、システムの基本データのアップデート方法についても、検討していきたいと考えている。

謝辞

また、本研究の一部は、科研費基盤研究 (B) 25280035 により行われた。ここに記して、謝意をあらわす。

[9]2015 年 10 月 21 日にアクセスして確認

参考文献

[1] Christian Bizer, Jens Lehmann, Georgi Kobilarov, Soren Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. DBpedia - a crystallization point for the web of data. *Web Semantics: Science, Services and Agents on the World Wide Web*, Vol. 7, No. 3, pp. 154 – 165, 2009.

[2] Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data - the story so far. *International Journal on Semantic Web and Information Systems*, Vol. 5, No. 3, pp. 1–22, 2009.

[3] 吉岡真治, Rhett Loban. Dbpedia の情報に基づく wikipedia のカテゴリ情報の一貫性の分析. 2015 年度人工知能学会全国大会 (第 29 回) 論文集, 2015. CD-ROM 1G4-2.

[4] Jim Giles. Internet encyclopaedias go head to head. *Nature*, Vol. 438, pp. 900–901, 2005.

[5] Besiki Stvilia, Les Gasser, Michael B. Twidale, and Linda C. Smith. A framework for information quality assessment. *Journal of the American Society for Information Science and Technology*, Vol. 58, No. 12, pp. 1720–1733, 2007.

[6] Meiqun Hu, Ee-Peng Lim, Aixin Sun, Hady Wirawan Lauw, and Ba-Quy Vuong. Measuring article quality in wikipedia: Models and evaluation. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, CIKM '07, pp. 243–252, New York, NY, USA, 2007. ACM.