検索エンジンを用いた類似文章検索システム EPCI の評価

田代 崇[†]　上田 高徳[†]　平手 勇宇[†,††]　山名 早人[†,‡]

† 早稲田大学基幹理工学研究科 〒169-85555 東京都新宿区大久保 3-4-1
†† 早稲田大学メディアネットワークセンター 〒169-85555 東京都新宿区大久保 3-4-1
‡ 国立情報学研究所 東京都千代田区一ツ橋 2-1-2

E-mail: †{ttashiro, ueda, hirate, yamana}@yama.info.waseda.ac.jp

あらまし 筆者らはこれまで，著作権侵害文章の抽出を目的として，既存の商用検索エンジンを用いた類似文章検索システム EPCI について提案してきた．EPCI システムは，以下の２つのステップによって構成される．（１）検索エンジンを用いて候補ページ集合を集めるステップ，（２）候補ページと入力文章との類似度に基づき候補ページ集合をランキングするステップ．候補ページの収集ステップでは，入力文章から文節を単位とした N-gram をクエリ集合として生成し，各クエリについて，候補ページ集合の平均類似度が閾値以下になるランキングまで上位から取得を行うことで，網羅性の向上を実現する．しかし，これまでの筆者らの研究では，ランキングの精度については評価してきたものの，候補ページ集合の網羅性についての評価が不十分であった．そこで本稿では，本手法の候補ページ収集ステップについての網羅性について評価を行った．その結果，文節の N-gram を利用した候補ページ収集は，文単位で区切ったようなクエリを利用した場合と比べて高い網羅性が得られることを確認した．

キーワード 類似文章検索，盗用検出，情報検索，検索システム評価

Evaluation of EPCI: Extracting Potentially Copyright Infringement texts by using a Search Engine

Takashi TASHIRO[†]　Takanori UEDA[†]　Yu HIRATE[†, ‡]　and　Hayato YAMANA[‡,‡]

† Graduate School of Fundamental Engineering, Waseda University 3-4-1 Okubo, Shinjuku-ku, Tokyo, 169-8555 Japan

†† Media Network Center, Waseda University 3-4-1 Okubo, Shinjuku-ku, Tokyo, 169-8555 Japan
‡ National Institute of Informatics 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430 Japan
E-mail: †{ttashiro, ueda, hirate, yamana}@yama.info.waseda.ac.jp

Abstract In our previous works, we proposed a new system for extracting similar texts from the web to extract copyright infringement texts called EPCI. EPCI consists of following two steps: (1) gathering candidate web pages by putting queries to a search engine, and (2) re-ranking candidate web pages with descending order of their similarity measurement to input texts. To achieve higher recall than previous related works, EPCI uses n-gram of chunks as queries. Though we confirmed that EPCI is able to extract similar texts with high precision of ranked candidate web pages, we have not evaluated recall. Therefore, in this paper, we evaluate recall of candidate web pages. As a result, we confirm that queries generated by n-gram of chunks provide higher recall than queries tokenized by sentence.

Keyword Similarity Search, Copy Detection, Information Retrieval, Information Retrieval Evaluation

1. はじめに

近年，ブログや Wiki といった環境の普及に伴って，誰でも容易に Web 上での情報発信が可能となった．このような環境によって，歌詞や新聞記事といった著作物の不正なコピーも，容易に Web 上に公開できるようになった．そのため，Web 上にある著作権侵害文章を発見することは，著作者の権利を守る上での重要な作業といえる． しかし，現在の Web ページの総数は 540 億[4]ともいわれており，Web 上のすべての Web ページを人手により調査し，著作権侵害文章を発見することは難しい．

大規模な文章集合から著作権侵害文章を発見するための補助となるシステムとして，類似文章検索シス

テムがある．類似文章検索システムを利用し，著作物と類似な文章を Web 上から抽出することができれば，効率的に Web 上の著作権侵害文章を発見することが可能となる．

類似文章検索を行う研究はこれまで盗用文章を発見することや，Web ページのクローリング時に重複したページの収集を避けることを目的として行われてきた．これらの既存の手法は，大きく分けて以下のように２つに分類できる．（１）文書 DB を独自に作成するもの．（２）検索エンジンをバックエンドに用いるものである．

文書 DB を作成するアプローチ[1][2][5][6]では，あらかじめすべての被検索対象の文章を取得し，インデックスを作成する．次に比較対象となる文章を入力し，当該文章と類似する文章を，インデックスを用いて効率的に抽出する．これらのアプローチでは，文書 DB を作成するためにストレージやインデックスを作成するための時間的コストが必要となる．そのため，Web を対象とした場合，膨大な Web データを対象にインデックスを作成できない．すなわち膨大なコストがかかるという問題があった．

一方で学生レポートが Web ページから盗用によって作成されていないかどうかを判定することを目的として，バックエンドに既存の商用検索エンジンを用いて類似文章を検索する手法も提案されている[7][8][12]．既存の検索エンジンを用いることにより，文書 DB を作成する手法と比べ容易に類似文章が検出可能である．基本的な流れは次の通りである．

（１）入力文章から，単語列や文，段落などの単位で，盗用の疑いのある部分を抽出する

（２）（１）で抽出した盗用の疑いのある部分を既存の商用検索エンジンにクエリとして投げ，検索結果が存在すれば，入力文章は盗用によって作成されたと判断する．この際，検索結果の文章をオリジナルの文章の候補としてユーザーに提示する．

しかし，これらの手法[7][8][12]は，Web に存在する盗用文章を網羅的に発見する目的には使用できない．これは，従来手法では，入力文章の盗用を判定するために，入力文章と類似した文章が Web 上に存在するか否かを判定するのにとどまっているからである．

そこで筆者らはこれまで，検索エンジンをバックエンドに用いて，できる限り多くの類似文章を Web 上から抽出するシステム EPCI を提案してきた[10][11]．EPCI システムでは，以下の３つのステップによって，Web 上に存在する類似文章を入力文章との類似性の高い順にユーザーに提示する．

（１）入力文章内から文節を単位とした N-gram を生成し，それを検索クエリ集合とする．

（２）検索結果の Web ページと入力文章の類似度の平均値が指定した閾値未満になるまで，ランキング上位から Web ページを取得し，候補ページ集合を作成する．この作業をクエリ毎に行う．

（３）各候補ページ集合内のページを対象に，入力文章との類似度を計算する．この際，候補ページ集合の URL から実際の Web ページを取得し，類似度を計算する．

提案手法に対して，歌詞や新聞記事といった著作物計 40 文章を入力として得られた文章のうち，[11]で定義する類似度 0.3 以上の文章を人手により確認したところ，１文章あたり平均 132.1 ページの類似文章を，平均 94.3%の精度で検出することを確認した[11]．

しかし，これまでの筆者らの研究では，網羅性についての評価が不十分であった．また，現実的な時間で検索結果を得るためには，少ない計算時間で出力を得なければならない．少ない計算時間で出力を得るためには，上記（２）において得られる，候補ページ総数を抑える必要がある．これは，候補ページ総数が多くなると，（３）における Web サーバへのアクセス回数と類似度の計算時間が増加するからである．そこで，本稿では文節 N-gram によるクエリ生成方法によって得られる候補ページ集合について，再現率，ページ総数を用いて，網羅性及び検索速度の検証を行った．

本論文の構成は以下の通りである．２節にて関連研究について述べ，３節ではこれまでの筆者らの提案手法について述べる．４節にて評価実験について述べ，５節にまとめを述べる．

2. 関連研究

以下では，従来の類似文章検索手法として，独自に文書 DB を作成する手法，及び検索エンジンを用いて類似文章検索を行う手法について紹介する．

2.1. 文書 DB を作成する手法

類似文章を抽出するための古典的な手法は，1995 年に S.Brin らによって提案された，COPS[1]と呼ばれるものである．COPS ではあらかじめ文章を文単位に区切り，ハッシュ表に登録しておく．次に入力文章と共通する文をもつ文書集合をハッシュ表から取得し，共通する文の数が閾値を超えた登録文章を出力する．また，SCAM[9]では，文章を文ではなく単語単位として，単語の転置インデックスを用いている．文より小さい単語単位で比較を行うことによって，文単位の改変が行われたような文章も抽出が可能となる．また，文献[6]では，Suffix-Tree をベースに部分文字列単位での検索が可能なシステムを提案している．

また，クローリング時の Web ページの重複を除去するために，近似的ではあるが高速に類似文章を検索す

る手法が提案されている．文献[2]では，２文章間で共通する N-gram の数に基づき類似文章抽出を行うが，このとき，各 N-gram のハッシュ値が低いものだけをインデックス化することで，インデックスのサイズを縮小する手法を提案している．また，文献[5]では，高次元の文書ベクトルを Chariker の提案する simhash[3]に変換することによって，インデックスのサイズを縮小する．Simhash のハミング距離は文書ベクトル間のコサイン類似度の近似値になるという特徴があり，simhash のみをインデックス化するだけで，類似文章の検索が可能である．

2.2. 検索エンジンを用いた盗用文章の判定手法

検索エンジンを利用して，入力文章と類似した文章を抽出する試みは，これまで学生レポート内に存在する，盗用文章を発見すること目的として行われてきた[7][8][12]．基本的な流れとしては，入力文章から盗用の疑いのある文や段落を抽出し，それを検索エンジンにクエリとして投げ，検索結果内の同一の文や段落を持つ文章を，入力文章が盗用したオリジナルの文章として出力している．文献[7]では１文内の単語数や接続詞に基づいて,文章の読みやすさを表す尺度を計算し，学生レポート内の盗用の疑いのあるパラグラフを抽出する．一方文献[8]では,「専門性が高い難解フレーズは盗用によって作成された可能性が高い」と仮定し，盗用部分を発見する．ここで「専門性の高さ」はフレーズ内の単語に含まれる文字数の多さによって判定している．文献[12]では，文章を文単位に区切り，文内に含まれる自立語の多い文をクエリとして投げ，検索件数に基づき盗用を発見する．

2.3. 関連研究のまとめと問題点

文書 DB を作成するアプローチでは，文書 DB の作成のために，あらかじめすべての被検索対象を独自に集め，インデックスを作成する必要がある．Web 上のすべての文章を被検索対象とした場合では，文書 DB の作成は，大規模なストレージと時間的なコストを必要とするため，適用が難しい．

一方検索エンジンを用いた場合，自前の文書 DB を作成するのに比べ,容易に類似文章の抽出が可能となる．しかし，従来の研究では，入力文章は盗用の疑いのある文章であり，その文章と類似したオリジナルの文章が Web 上に存在するかどうかを判定するだけにとどまっている．本研究のように Web 上にある盗用文章を発見するための類似文章検索システムでは，オリジナル文章を入力として Web 上にある盗用文章を類似文章として抽出する必要がある．そのため，従来手法では，以下のような問題があった．

（１）盗用される場所をオリジナル文章から判断することは不可能であること．

（２）オリジナル文章はどこをコピーされても盗用となり，文章の一部分から切り出したクエリでは，十分な網羅性を得られないこと．

3. 検索エンジンを用いた類似文章検索手法

本節では，これまで筆者らがこれまで提案したシステム EPCI[10][11]について述べる．以下， 3.1 で EPCI システム[10][11]が扱う類似文章について述べ, 対象とする Web ページを定義する. 3.2 では EPCI システムの概要について述べる．3.3 では提案システムの具体的な構成について述べる．

3.1. 対象とする Web ページ

本手法が抽出対象とする文章とは，入力文章と類似した文章を一部または全体にもつ Web ページである．

また一般に,「類似文章」は，大きく分けて以下の２つに分類できる．

（１） 深層的な類似文章

入力文章と同じ事柄に対して異なった記者が書いた新聞記事や,同じテーマにそって書かれた文章など，意味的に一致している文章．

（２）表層的な類似文章

入力文章をコピーアンドペーストした文章など，文章そのものが類似した文章である.軽微な語尾変化や，漢字とひらがな間の変換など，改変が伴ってコピーされた文章も,（２）に分類される．

本分類において，本手法が対象とする類似文章とは,（２）の表層的な文章のことを指す．

3.2. EPCI システムの概要

EPCI システムでは，検索エンジンを用いて，以下の３つのステップにより，できる限り多くの Web 上にある類似文章を抽出する．

（１） 入力文章内のすべての文節を単位とした N-gram からなる検索クエリ集合を生成する．

（２） すべてのクエリについて，検索結果の Web ページと入力文章の類似度が閾値 *th* 未満になるまで，ランキング上位から M 件ずつ Web ページを取得し，候補ページ集合を作成する．

（３） 候補ページ集合を入力文章との類似度を計算する．

入力文章全体から文節単位のＮ-gram をクエリとすることで，理論的には被検索対象内に連続する N 文節が一致していれば抽出が可能となる．また，網羅性を向上させるためには，作成されたクエリに対して，上位何件まで取得するかも重要な問題となる．これに対して本手法では，各クエリに対して，平均類似度が一定値を超えるまで取得することにした．

3.3. EPCI システム

類似文章の検索のおおまかな流れを，図１に示す．入力された文章はまず，文章解析によって文節列に分割される．次にクエリ生成では，文節列からクエリ集合を生成する．候補ページ取得では，生成された各クエリに対して，検索エンジン API を利用して URL のリストを取得した後，各 URL に対して Web サーバから実際の Web ページを取得する．最後に入力文章から生成された文節列と得られた Web ページ間の類似度を計算し，計算したものから逐次的にユーザーに提示する．また，収集した候補ページ集合の平均類似度の高いクエリについては，再度候補ページ集合の取得を行う．

以下では，クエリの生成，候補ページ取得，類似度の計算について述べる．

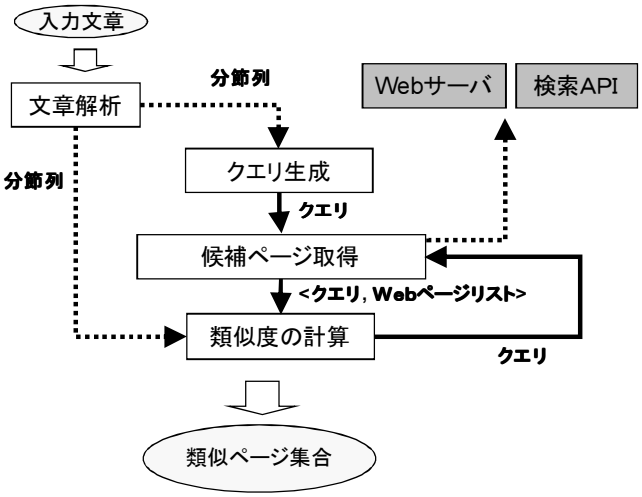

図１．EPCI システムにおける類似文抽出の流れ

3.3.1. 検索クエリ生成

できる限り多くの類似文章を抽出するための検索クエリの要件は，以下の２点である．

（１）検索結果を絞り込むこと

（２）改変された文章を抽出すること

（１）が重要となるのは，既存の商用検索エンジンでは，表２に示すように，各クエリに対して最大でも MSN では上位 250 件，Google や Yahoo!では上位１０００件までしか取得することができないからである．検索結果が絞り込めないクエリでは，取得可能なランキング内に目的のページを集めることができなくなり，精度だけでなく網羅性も下がる．

したがって，（１），（２）の点から，網羅性の高い理想的な検索クエリとは，改変された部分を含まない部分によって構成され，なおかつ検索結果が絞り込めるクエリである．しかし，盗用された文章は複数存在し,複数の改変パターンが存在すると考えられるため、文章のどの部分が改変されるかを，一概に特定することは難しい．

そこで，まず入力文章を文節列に分解し，文節列から文節単位の N-gram, すなわち入力文章内のすべての連続する N 個の文節，をクエリとすることを考えた．１つの文節ではなく，連続した N 文節の検索クエリを実現することで，１文節で検索するよりも検索結果を絞り込める．また理論的には，連続する N 文節が抽出対象とする Web ページ内に存在すれば，抽出が可能となる．

N=2 とした場合の例を表１に示す．なお本生成方法では，同じクエリが複数生成される場合があるが，重複したクエリはすべて除去を行ったものが，最終的なクエリ集合となる

表１．検索クエリの生成例

文節列	“今日は””公園で””サッカーを”“しました”
クエリ１	“今日は”“公園で”
クエリ２	“公園で”“サッカーを”
クエリ３	“サッカーを”“しました”

3.3.2. 候補ページの取得

候補ページの取得では，クエリ生成によって作られたクエリと,類似度の計算後に再入力されたクエリの，２つのパターンのクエリを入力とする．これらのクエリに対して行う処理は以下の２つである．

（１）　検索エンジン API にクエリをなげ，M 件の URL リストを取得する．
この際，クエリが初めて検索するものであれば，検索結果１位から M 位までの URL リストを取得する．再入力されたクエリであれば，そのクエリが現在まで取得したランキング位置から下位 M 件の URL リストを取得する．

（２）　URL リストから，実際にその URL に存在する Web ページを取得する．
一度取得したページはキャッシュし，すでにページを取得済みの URL の場合,キャッシュから Web ページを取得する．

ここで,Mは検索エンジンから一度に取得する検索結果数である．M の最大値は，表２に示すように検索エ

表２．主要３社の商用検索エンジン API の仕様

	Yahoo!JP[17]	MSN[16]	Google[13]
アクセス方法	REST	SOAP	SOAP
検索制限回数	50,000 回	10,000 回	1000 回
取得可能な検索結果数	100	50	10
公開日		**2005/9**	**2002/4**
1クエリ内の最大検索語数	**なし**	**なし**	**32**

ンジンごとに定められている．

3.3.3. 類似度の計算

類似度の計算ステップでは，候補 Web ページと入力文章との類似度を計算する．

類似度の計算は，次のステップにより計算される．

（１）各候補 Web ページに対して，入力文章と共通の文節を抽出し，抽出された文節の出現順序を保持した文節列を作成する．

（２）入力文章の文節列と（１）で抽出した文節列に対して，以下に示す LCS 類似度を計算する．

LCS 類似度

LCS 類似度は，入力文章全体と，Web ページの一部分の類似性を，文節の出現順序の類似性に基づいて定量化したものである．

入力文章の文節列を S，候補 Web ページの文節列を C とすると，LCS 類似度は以下の(a)式により定義される．

$$Sim(S,C) = \log_2\left(\frac{|Lcs(S,C)|}{|S|}+1\right) \quad \cdots \quad (1)$$

入力文章が候補 Web ページに完全に含まれている場合，$Sim(S,C)=1$ となる．一方で，全く含まれていない場合 $Sim(S,C)=0$ となる．ここで，$Lcs(S,C)$は，２つの列 S, C 最長共通部分列を表す．最長共通部分列は，２つの列に共通に含まれる部分列の中で最長なものであり，本節最後に定義を示す．

文節の LCS を利用すると，コメントなどの入力文章とは関係のない文が，文章間に挿入される前後で類似度は変わらないという利点がある．

最長共通部分列(LCS)

列 $A=<a_0, a_1, .., a_{N-1}>$, $B=<b_0, b_1, .., b_{M-1}>$ について，以下にあげる３つの条件を満たすように，インデックスの列$<i_0, i_1, \ldots, i_{L-1}>$，$<j_0, j_1, \ldots, j_{L-1}>$ を選んだとき，$<a_{i_0}, a_{i_1}, \ldots, a_{i_{L-1}}>$，（または$<b_{j_0}, b_{j_1}, \ldots, b_{j_{L-1}}>$）を共通部分列という．

(1) $a_{i_0} = b_{j_0}, a_{i_1} = b_{j_1}, \ldots, a_{i_{L-1}} = b_{j_{L-1}}$

(2) $0 \le i_0 < i_1 < \ldots < i_{L-1} \le N-1$

(3) $0 \le j_0 < j_1 < \ldots < j_{L-1} \le M-1$

列 A,B の最長共通部分列（ＬＣＳ）とは，列 A,B の共通部分列の中で，最も長い列である．

4. 評価実験

本節では，提案システムの網羅性に関する評価について報告する．本節では，まず 4.1 にて実験に利用した入力文章について述べ，4.2 にて精度に関する評価を行った実験[11]を簡単に説明する．最後に 4.3 にて，網羅性に関する評価実験について述べる．

4.1. 入力文章

実験にあたり入力文章として，英語新聞記事，英語歌詞，日本語新聞記事，日本語歌詞の 4 種類のカテゴリから 10 文章（以下，シードと呼ぶ）ずつ計 40 シードを下記の 4 サイトから選出した．

（１） Yahoo! News[18]の 2006 年 11 月 10 日の閲覧数ランキングから上位 10 シード

（２） Lyrics Search Engine[14] の 2006 年 11 月 16 日の閲覧数ランキングから上位 10 シード

（３） goo ニュース[19]の 2006 年 11 月 10 日の閲覧数ランキングから上位 10 シード

（４） 歌詞 GET![1]の 2006 年 11 月 16 日の Daily 閲覧数ランキングから上位 10 シード

選択した入力文章の文の数，及び，語数をまとめたものを表３に示す．文については，新聞記事の場合にピリオドからピリオドまでを，歌詞の場合は１フレーズ(掲載サイトの掲載形式の１行)を文として定義し，人手により分割した．また，語数は，日本語の場合文字数であり，英語の場合単語数を表す．

表３.評価に利用した入力文章

	文章 ID	平均文数	平均語数
日本語新聞	1~10	54.4	1,400
日本語歌詞	11~20	9.10	1,110
英語新聞	21~30	31.6	460
英語歌詞	31~40	88.8	451

4.2. 精度の評価

提案システムによって得られる文書集合を，実際に人手により閲覧し，提案システムの精度について評価をおこなった．具体的には，提案システムに 4.1 に示す入力文章を入力として得られる最終的な検索結果のうち，LCS 類似度 0.3 以上のものを実際に人手により閲覧し，表４に示すようなスコア付けを行った．このとき，検索エンジンは Yahoo! Japan を利用し，検索を打ち切る平均類似度 th は 0.2，一度に検索エンジンから取得する件数 M は 20，検索クエリは文節 3-gram を利用した．表５は，得点別に得られたページ数を分類したものである．何がしらの類似部分が含まれる得点１以上の文章を正解と定義すると，LCS 類似度 0.3 以上の精度は約 94.0%である．

表４．文章の得点の定義

	ページに含まれる入力文章のコピーの割合（人手による判断）
得点 3	80%以上
得点 2	30% から 80%
得点 1	30%未満
得点 0	0% (関係のないページ)

表５．各得点における類似度 0.3 以上のページ数

	得点 3	得点 2	得点 1	得点 0	合計
英語新聞	1,658	334	26	28	2,046
英語歌詞	1,075	272	16	160	1,523
日本語新聞	213	68	19	72	372
日本語歌詞	1,240	354	36	62	1,692
合計	4186	1028	97	322	5,633

4.3. 網羅性と検索速度の評価

本項では，文節のＮ-gram をクエリとした場合の候補ページ集合と，文をクエリとした場合の候補ページ集合の再現率の比較を行うことで網羅性を評価する．また，候補ページ集合で網羅性とトレードオフとなる検索速度についての評価も行う．候補ページ集合の精度については，評価対象としない．これは，EPCI システムでは集められた候補ページ集合は、最終的 LCS 類似度に基づきランキング付けされるため，候補ページ集合の精度が悪くても，最終的に得られるランキングの精度は変わらないからである．

検索速度の検証では，クエリタイプごとに候補ページ総数の比較を行った．これは，候補ページ総数が増加すると，類似度を計算する際に Web サーバへのアクセス回数が増加し，EPCI において最もボトルネックとなるからである．また，実際に候補ページ URL を集める際にも，API を通じて Web サーバへのアクセスが生じる．しかし，Yahoo!や Google といった商用検索エンジンでは通常のサイトと比べるとレスポンスが早く，また一度に 10 件から 100 件の検索結果が取得可能であるため，候補ページ総数に比べて検索速度への影響が少ない．そのため，評価対象としなかった．

4.3.1. 評価方法

比較を行うクエリタイプ，再現率について定義する．

クエリタイプ

以下の 6 種のクエリタイプを比較する．

（１） 文
（２） 文のフレーズ検索（文を’”’で囲ったもの）
（３） 文節 2-gram
（４） 文節 3-gram
（５） 文節 4-gram
（６） 文節 5-gram

再現率

評価にあたり，0.3 以上の LCS 類似度をもつ候補ページを正解ページと定義した．また比較対象としたすべてのクエリタイプ得られる全正解ページの和集合を正解ページ集合とした．

入力文章 s から生成されたクエリ集合を $query\_set_s$，検索を終了する平均類似度の閾値 th としたときの再現率を(2)式で表す．

$$RECALL_M(query\_set_s, th) = \frac{|R_M(query\_set_s, th)|}{\left|\bigcup_{Q_s \in all\_query\_set_s} R_M(Q_s, 0.0)\right|} \quad \cdots (2)$$

ここで，$R_M(query_set_s, th)$は $query\_set_s$ 内の各クエリに対して，平均類似度が th 未満となるまで取得した場合に得られる正解ページの集合である．また，M は一度に検索エンジンから取得する候補ページ数を表し，$all\_query\_set_s$ は比較対象となる６種のクエリ集合を要素とする集合である．

$R_M(query_set_s, th)$は th=1.0 のとき最小となり，th=0.0 のとき最大となる．とくに th=0.0 のときは，すべてのクエリに対して，取得可能な最大数の検索結果を検索エンジンから取得した場合であり，このときの $RECALL_M(query_set_s, 0.0)$を $query\_set_s$ の**最大再現率**と呼ぶことにする．例えば，Yahoo! Japan で検索を行った場合，$R_M(query_set_s, 0.0)$は $query\_set_s$ のすべてのクエリについて，最大上位 1000 件までの候補ページを取得して，得られるすべての正解ページをマージしたページ集合を指す．また，本実験では，検索制限回数が多い Yahoo!Japan の検索エンジンを用い，M は Yahoo! Japan の最大値である 100 とした．

フリードマン検定と対比較

本実験ではクエリタイプ間で再現率や候補ページ総数に差異があるかどうかを統計的に判定するために，フリードマン検定とシェッフェの方法による対比較をおこなった．フリードマン検定ではすべてのクエリタイプ間で最大再現率や候補ページ総数の差異があるかどうかを検定する．フリードマン検定において差異が認められた場合において，具体的に２つのクエリタイプで比較を行うものが対比較である．対比較では，一つの入力文章ごとに，クエリタイプごとに最大再現率の降順（または候補ページ総数昇順）で順位付けを行う．この際，同一の順位が存在した場合，順位の平均[1]がその手法の順位となる．最後にすべての入力文章で得られる順位の平均値を平均順位として求める．クエリタイプ A がクエリタイプ B より平均順位が低く，A,B 間の有意確率が有意水準 p 未満である場合，手法 A は手法 B より高い再現率（または低い候補ページ総数）が得られることを表す．本実験では，p=0.05 とした．

4.3.2. 実験結果

最大再現率の比較

各クエリタイプにおける最大再現率の比較結果を表６，最大再現率の分布を図２に示す．

[1] Ｎ位から N+K 位が同一再現率（もしくは候補ページ数）であれば，Ｎ位から K 個のクエリタイプの平均順位はすべて，(2N+K)/K となる．

表６，図２によると，クエリタイプ間の比較では，文節 N-gram によるクエリは，N=2,3,4,5 のいずれの場合においても，文や文フレーズをクエリとして用いた場合と比べて平均順位，最大再現率が高い．また，対比較を行った結果においても，文節 N-gram を利用したクエリは，文や文フレーズを用いたクエリよりも，高い再現率を得られることが確認できる．また，文節 N-gram 間の比較では，文節 3-gram が最も高く，ついで文節 4-gram が高かった．しかし，文節 3-gram と 5-gram 間で統計的に差異が認められたものの，それ以外には統計的な差異を確認することができなかった．

表６．最大再現率の比較

	平均順位	最大再現率（平均）	差異があるもの
文	5.1	0.361	文節 N-gram (N=2,3,4,5)
文フレーズ	5.4	0.319	文節 N-gram (N=2,3,4,5)
文節 2-gram	2.9	0.626	文,文フレーズ
文節 3-gram	1.6	0.717	文,文フレーズ, 文節 5-gram
文節 4-gram	2.1	0.673	文,文フレーズ
文節 5-gram	3.0	0.621	文,文フレーズ

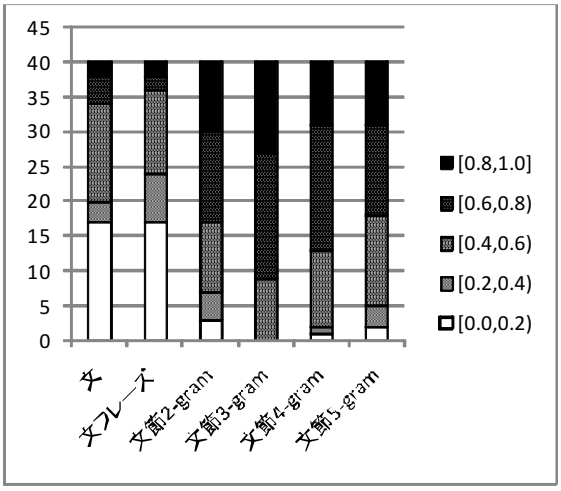

図２．クエリタイプ捌最大再現率の分布

文や文フレーズによるクエリが文節 N-gram と比べて再現率が悪くなるのは，クエリ内に改変された語が含まれやすく，改変された文章が抽出できないためであるといえる．一方で文節 N-gram 間で大きな差異が得られないのは，入力文章によって適切な N が異なるからであると考えられる．例えば，特徴的な文節が多い文章では,N が低くても検索結果を絞り込めるため，改変された文節を含みにくいNが低いクエリが適しているといえる.一方で特徴的な文節が少ない文章では，検索結果が絞り込める N が高いクエリが適している．

候補ページ数の比較

同一の再現率であるならば，少ない候補ページ総数で抽出できることが望ましい．そこで，閾値 *th* を変化させ，再現率と、そのときに取得した候補ページ総数を求め，同一の再現率においての候補ページ総数の比較を行った．入力文章によっては，指定した再現率に達していないクエリタイプがある．そのため，本実験では，すべてのクエリタイプで指定した再現率を実現できた入力文章のみ，対象とした．

表８，表９では，各再現率における，候補ページ数に基づいたクエリタイプ間の平均順位と，平均候補ページ数を表す．どの再現率においても，文フレーズがもっとも候補ページ総数が低い．また，文節 N-gram

表８．各再現率における候補ページ数の平均順位

再現率	対象文章数	有意確率	クエリタイプごとの平均順位					
			文	文フレーズ	文節 N-gram			
					N=2	N=3	N=4	N=5
0.1	30	1.21E-23	3.1	1.4	5.9	4.8	3.5	2.3
0.2	19	3.96E-15	2.9	1.4	6.0	4.8	3.5	2.3
0.3	17	5.66E-12	2.9	1.5	5.9	4.9	3.2	2.6
0.4	14	1.94E-10	2.6	1.6	6.0	4.9	3.4	2.6
0.5	8	3.41E-05	3.1	1.3	5.8	4.6	3.6	2.6

表９．各再現率における候補ページ数（すべての対象文章の平均）

再現率	対象文章数	文	クエリタイプごとの候補ページ総数（平均値）				
			文フレーズ	文節 N-gram			
				N=2	N=3	N=4	N=5
0.1	30	4027	432	3712	1877	1076	681
0.2	19	1015	440	5052	2454	1306	821
0.3	17	1615	913	6369	3319	1814	1350
0.4	14	2038	1178	8415	4190	2365	1691
0.5	8	7966	1391	12640	9409	4836	3102

表１０．各再現率における差異が認められるペア

	候補ページ少ない	候補ページ多い
0.1	文	文節 N-gram(N=2,3)
	文フレーズ	文，文節 N-gram(N=2,3,4)
	文節 4-gram	文節 2-gram
	文節 5-gram	文節 N-gram(N=2,3)
0.3	文	文節 2-gram
	文フレーズ	文節 N-gram(N=2,3)
	文節 4-gram	文節 2-gram
	文節 5-gram	文節 N-gram(N=2,3)
0.5	文フレーズ	文節 N-gram(N=2,3)
	文 5-gram	文節 N-gram(N=2)

間ではＮが小さくなるにつれて候補ページ総数が多くなる．文の場合では，概して文節 5-gram より多く，文節 4-gram よりも少ない候補ページ数である．表１０では，再現率 0.1,0.3,0.5 において，クエリタイプ間で対比較を行った結果を示す．ここにおいても，文フレーズが最も低い候補ページ数であることが統計的に確認できる．再現率が高くなるにつれてクエリ間の差異が確認できなくなっているが，これは，対象文章数が少なくなり，差異を確認するには統計的に十分な入力数がないからと考えられる．

実験のまとめと考察

以上の実験結果をまとめる．文や文フレーズ，文節 N-gram でＮが大きい(5 以上)場合による検索クエリは，再現率が低いことを許容できる場合に利用するのが，検索速度の観点から望ましい．例えば，入力文章の盗用を発見する目的にはオリジナル文章を発見できれば良く，高い再現率はいらないため，検索速度が速い文や文フレーズ, 文節 5-gram をクエリとするのが適しているといえる．一方，文節 3-gram,4-gram などの N が小さい場合のクエリは，検索速度は文よりも遅くなるが, 高い再現率を得ることが可能である．例えば, Web 上に存在する著作権侵害文章を集めるためには，高い再現率が必要となるため，文節 N-gram の方が文をクエリとするよりも適しているといえる．

5. まとめ

本論文では，これまで筆者らが提案してきた検索エンジンを用いた類似文章検索システム EPCI[10][11]について，網羅性と検索速度についての評価を行った．その結果，クエリ数については，文をクエリとした場合と比べて劣るものの，高い網羅性で取得可能であることを確認した．このように網羅性の高い類似文章検索システムは，Web 上に存在する著作権侵害ページを抽出するためには有効なシステムといえる．

謝辞

本研究の一部は文部科学省リーディングプロジェクト「e-Society」及び科研(特定)「情報爆発に対応する高度にスケーラブルなモニタリングアーキテクチャ」によるものである．

文　献

[1] S. Brin, J. Davis, and H. Carcia-Molina: ” Copy detection mechanisms for digital documents,” In Proc. of the ACM SIGMOD Annual Conference, pp.398-409, 1995.

[2] A. Z. Broder, S. C. Glassman, M. S. Manasse and G. Zweig, "Syntactic Clustering of the Web", In Proc. of Int'l World Wide Web Conference, pp.291-303, 1997

[3] M. S. Charikar, "Similarity Estimation Techniques from Rounding Algorithms", In Proc. of the Annual ACM Symposium on Theory of Computing, pp.380-388, 2002

[4] Y. Hirate, S. Kato and H. Yamana: “Web Structure in 2005,” In Proc. of WAW2006, 2006 .

[5] G. S. Manku, Jain, A. D. Sarma: "Detecting Near-Duplicates for Web Crawling ", In Proc. of Int'l World Wide Web Conference, pp.141-151, 2007

[6] K. Monostori, A. Zaslavsky, and H. Schmidt: “MatchDetectReveal: Finding Overlapping and Similar Digital Documents”, In Proc. of IRMA Conference 2000, pp.955-957, 2000.

[7] C. J. Neill, G. Shanmuganthan: "A Web-Enabled Plagiarism Detection Tool," IT Professional, Vol. 6, No. 5, pp. 19-23, 2004.

[8] S. Niezgoda, T. P. Way: “SNITCH:A Software Tool for Detecting Cut and Paste Plagiarism”, In Proc. of SIGCSE Technical Symposium on Computer science education ,pp.51-55, 2006.

[9] N. Shivakumar and H. Carcia-Molina: “ SCAM: A Copy detection mechanisms for digital documents,” In Proc. of 2nd Int'l Conference on the Theory and Practice of Digital Libraries, 1995.

[10] T.Tashiro, T. Ueda, T. Hori, Y. Hirate, H. Yamana: “Copyright Violation Detection System for Web Texts”, DBSJ Letters, Vol5, No.2, pp.25-28, 2006

[11] T.Tashiro, T. Ueda, T. Hori, Y. Hirate, H. Yamana: “EPCI: Extracting Potentially Copyright Infringement texts from the Web”, In Proc. of Int’l World Wide Web Conference, pp.1151-1152, 2007

[12] Liu, Yi-Thing Zhang, Heng-Rui Chen, Tai-Wei Teng, Wei-Guang: “Extending Web Search for Online Plagiarism Detection”, In Proc. of IEEE Int’l Conf on Information Reuse and Integration(IRI 2007), pp.164-169, 2007.

[13] Google: http://google.com.

[14] Lyrics Search Engine: http://lyrics.astraweb.com.

[15] Mecab: Yet Another Part-of-Speech and Morphological Analyzer:http://mecab.sourceforge.net/

[16] MSN Search: http://msn.com.

[17] Yahoo! Japan: http://yahoo.co.jp.

[18] Yahoo! NEWS: http://news.yahoo.com.

[19] goo ニュース: http://news.goo.ne.jp.

[20] 歌詞検索エンジン　歌詞ゲット!: http://www2.kget.jp.