Vol.2011-CE-109 No.4
2011/3/18

Web 検索 API を用いた 9 ヶ国語作文支援ツール

古川陽平[†]　　安藤一秋[†]

近年，英語の論文作成や中国語でビジネスメールのやりとりなど第二言語で作文する機会が増加している．一般的に，第二言語で作文する際に辞書を用いるが，辞書には少数の一般的な用例しか存在しないため，書きたい文の参考にできない，複数の訳語候補から最適な語が選択しにくいなどの問題がある．そこで，検索エンジンを利用した作文支援が注目されている．Web 上に存在する多言語テキストデータを用例文と仮定し，検索エンジンで得られる用例文や Web 上の利用頻度であるヒット数は作文時に有用な情報となる．本研究では，検索エンジンを利用して，多言語での作文を支援するシステムを構築中である．本ツールでは 9 カ国の言語に対し，ヒット数による表現の汎用性・妥当性の検証，文中の誤り候補の検出，Web からの用例文の自動抽出など,作文に有用な情報を提示する．本論文では，誤り候補検出の改良として，ヒット数差を用いた誤り候補検出手法を提案する．また，誤り検出機能の性能評価について述べ，有用性を示す．

A Composition Support Tool for 9 Languages using Web Search API

YOHEI FURUKAWA[†]　KAZUAKI ANDO[†]

In recent years, the opportunities to read and write a sentence in second language have been increasing. When we write a sentence in second language, we consult dictionaries to find words and example sentences. However, we often cannot choose the suitable word, and find example sentences which we want in them. Therefore, composition support using a search engine is attracting attention. Since the Web is an immense multilingual text data, example sentences and the hit count which are obtained by a search engine are useful information for writing a sentence. In this research, we are developing a support system for multilingual composition using a search engine. The tool can provide useful information for writing a sentence in 9 languages. Using this tool, we support verification of the validity of words based on the hit count, detection of mistake candidates in a sentence, extraction of example sentences from the Web, and so on. In this paper, we propose a new error detection method using hit count difference. Finally, this paper shows the effectiveness of the proposed error detection method.

1. はじめに

近年，英語の論文作成や中国語でのビジネスメールのやりとりなど，第二言語で作文する機会が増加している．それに伴い，誤りが無いことはもちろん自然な文での作文が要求されるようになった．一般的に，第二言語で作文する際に辞書を用いるが，辞書には少数の一般的な用例しか掲載されないため，書きたい文の参考にできない，複数の訳語候補から最適な語が選択しにくいなどの問題がある．また，辞書だけでは，作成文の誤りを発見しにくい，自然な文であるのか判定できないなどの問題もある．

そこで，Web 上に存在する多言語テキストデータを用例文と仮定し，検索エンジンを用いて得られる用例文やヒット数を作文支援に活用する研究が注目されている．検索エンジンで得られる多くの用例文を参照することで，書きたい文に対する表現や言い回しの参考にできる．また，ヒット数は Web 上での利用頻度とみなせるため，実際に利用される自然な訳語や表現の選択・判断基準となる．

本研究ではこれらの特徴を生かし，第二言語で作文するユーザに対し，検索エンジンによって得られる Web 上の多言語テキストデータを用いて作文時に有用な情報を提示し，ユーザの試行錯誤を支援するツールの実現を目的とする．

Web 上のテキストデータを利用した作文支援 [1)-6)]の例として，大鹿らの英作文支援システム [1)]や平野らの英文冠詞誤りの検出 [2)]などがある．大鹿らの研究では，検索エンジンで得られるヒット数や用例文を用いて，語・表現の妥当性検証や可読性を加味した用例提示を行う．平野らの研究では，入力文に対し不要単語の削除を行い，冠詞の誤り検出に特化したフレーズのヒット数を利用して誤り検出を支援する．また，作文支援の関連ツール [7),8)]の例として，NativeChecker[7)]などがある．NativeChecker では，検索エンジンを用いて，最適な前置詞や類義語などの検証，用例提示を行う．しかし，これらの関連研究・ツールは単一言語のみを対象としたものである．Web 上には多言語テキストデータが存在し，検索エンジンを用いることで多言語を扱うことが可能である．また，第二言語で作文する際には，検証支援や誤り検出など単独の支援ではなく，個々のユーザニーズに対応できる総合支援が必要となる．

そこで我々の先行研究（先行ツール）では，検索エンジンで取得できる Web 上の多言語テキストデータを利用し，9 カ国の言語（日本語・英語・ドイツ語・中国語・韓国語・フランス語・イタリア語・スペイン語・ポルトガル語）に対応した作文支援ツールを構築した．先行ツールには，言語共通の支援機能と各言語特有の資源や解析ツールを活用することで高度な支援を実現する言語に特化した機能を備えている．言語共通の支援機能には，文中の誤り候補を検出する *n*-gram 検索機能や検出した誤り候補

[†] 香川大学工学部
Faculty of Engineering Kagawa University

の訂正を支援する機能がある．*n*-gram 検索機能では，入力文を *n*-gram 系列に分割して Web 検索し，各系列のヒット数を用いた誤り候補検出を行う．しかし，*n*-gram 系列単独のヒット数のみを用いて誤り候補を検出すると，フレーズが長い系列を誤検出する，検出した長いフレーズ中のどこが誤りなのかが判断できない問題がある．

そこで本稿では，先行ツールの *n*-gram 検索の改良として，*n*-gram 系列のヒット数差を用いた誤り候補検出手法を提案する．誤り候補検出の提案手法では，*n*-gram 系列のヒット数差を用いてスコア計算し，スコアが高い系列を誤り候補として検出する．また，スコアを用いて 2-gram まで誤り箇所を絞り込むこともできる．

なお，本ツールは，作文対象言語の基本文法を一通り学習した中級レベルのユーザを対象とする．また，Web アクセスには，Yahoo! API[9]を利用している．

2. 関連研究・ツール

Web 上のテキストデータを作文支援に活用する多くの研究 [1)-6)]がなされている．例えば，訳語・表現の妥当性検証や用例検索などの総合的な作文の支援に関する研究[1)-3)]，冠詞や前置詞などの誤り検出・訂正に関する研究[4)-6)]がある．

総合的な作文の支援に関する研究例として，大鹿らの研究[1)]がある．この研究では，Google API を用いて，訳語・表現の妥当性を検証する機能や snippet を利用した用例提示機能を有した英作文支援システムを提案している．検索機能は本研究と類似しているが，提示方法や用例の利用法などが異なる．

誤り検出・訂正に関する研究例として，平野らの研究[2)]がある．この研究では，検索エンジンのヒット数を活用した英文冠詞誤り検出を提案している．具体的には，入力文から不要な語を削除することでヒット数を得やすくし，単数形・複数形や全冠詞パターンを網羅的に検索することで，高精度の誤り検出を実現している．

また，本研究に関連するツール[7),8)]がいくつかある．例えば，NativeChecker[7)]では，検索エンジンで得られるヒット数を用いて，英文の最適な前置詞や類義語，活用形などの検証が行える．また，検索エンジンの snippet を利用した任意フレーズの用例も参照できる．

これらの関連研究・ツールでは，単一言語を対象としたものが多い．また，語・表現の妥当性検証や誤り検出，用例検索など個々の機能に特化したものが多い．しかし，Web 上の多言語テキストデータを利用することで，多言語に対応した作文支援が可能である．また，第二言語で作文する際には，検証支援や誤り検出など単独の支援ではなく，個々のユーザニーズに対応できる総合支援が必要となる．

本稿で提案する作文支援ツールでは，9 ヶ国語に対してヒット数や用例文などを活用し，最適な語・表現の選択や作成文の正誤判定および誤り位置の特定，用例提示など，多言語作文時におけるユーザの試行錯誤を総合的に支援する．

3. 多言語作文支援ツール

3.1 ツールの構成

提案するツールの構成を図 1 に示す．

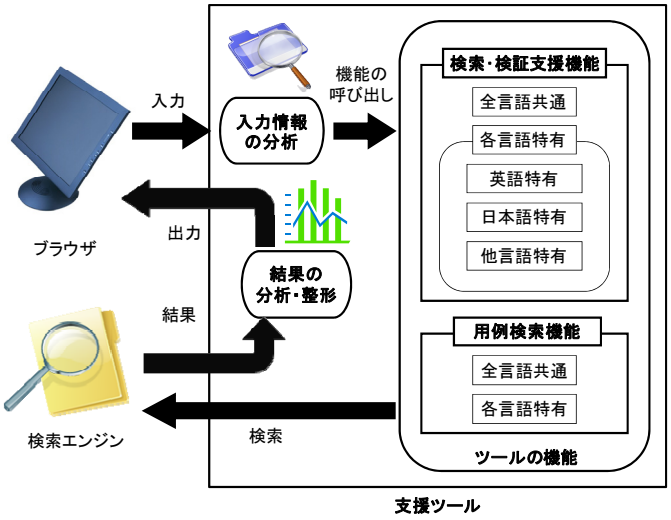

図 1　ツールの全体構成

図 1 に示すように，ブラウザから入力された情報を分析した後，必要な機能を決定し，各機能へデータを送る．送られたデータは，全言語共通モジュールで処理された後，言語タイプに応じて各言語特有のモジュールで処理される．検索・検証支援機能の全言語共通モジュールは，ブラウザから入力された情報の中から検索式の定義に必要な情報のみ抽出し，その情報を基に検索式を自動生成する．そして，Yahoo! API を用いて Web にアクセスし，取得した結果からヒット数や用例文など検索・検証支援に必要な情報を抽出する．検索・検証支援機能の言語特有のモジュールでは，共通モジュールが抽出した情報に対して，各言語特有の資源や解析ツールを利用して，単語・熟語レベルの判定や係り受け解析などを行う．用例提示機能では，まず，全言語共通モジュールが検索・検証支援機能で処理した情報を受け取り，用例を抽出する．用例提示機能の各言語特有モジュールでは，各言語特有の資源を利用し，ユーザの語彙レベルに応じた用例の並び替えや熟語の提示などを行う．検索・検証支援機能および用例提示機能の出力は，グラフや表など，ユーザが参照しやすいように整形された後，ブラウザに表示される．

3.2 ツールのインタフェース

本ツールでは，日本語・英語・ドイツ語・中国語・韓国語・フランス語・イタリア語・スペイン語・ポルトガル語の 9 ヶ国語に対応し，それぞれのインタフェースを備えている．例として，日本語のインタフェースを図 2 に示す．インタフェースの上部は入力部分であり，下部は出力部分である．

入力部分では，“Ingerface Language” を選択することで各言語専用のインタフェースに切り替わる．また，“入力言語” を選択することで作文対象（検索用）言語が設定できる．n-gram 検索や domain 検索などの各検索機能はボタン表示されており，クリックすることで選択できる（Default ではフレーズ検索となる）．また，選択機能に応じて設定項目が動的に提示されるため，ユーザは検索意図に応じた設定が簡単に行える．

出力部分では，選択機能に応じた結果を表示する．入力と出力を同一ページに表示することで，出力結果を参考にしながら入力（検索語）を変更するなどの試行錯誤が容易になる．

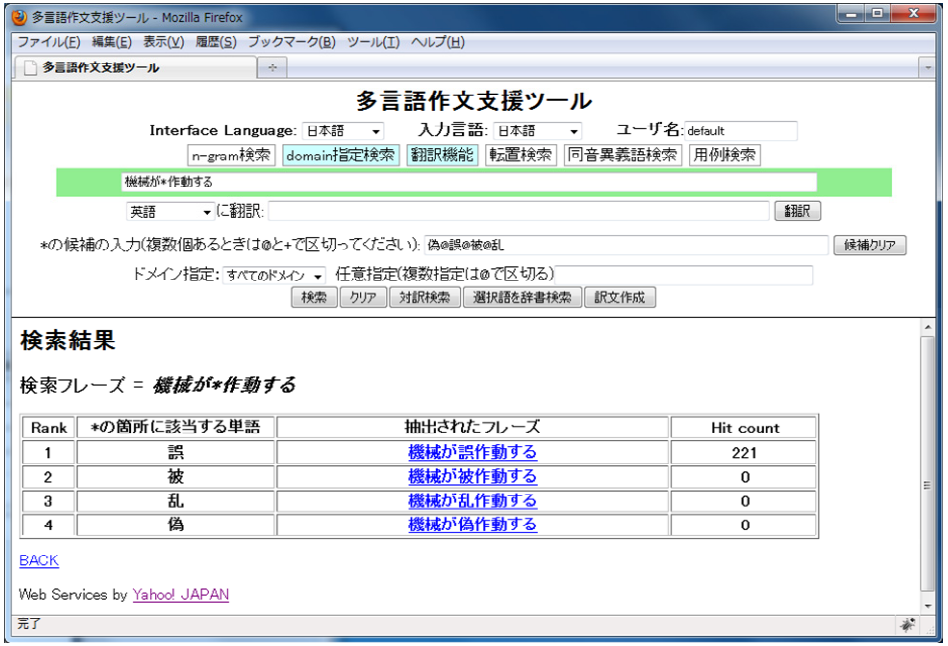

図 2　日本語のインタフェース

3.3 検索・検証支援機能

検索・検証支援として，語・表現の妥当性検証を支援する機能，文中の誤り候補を検出・訂正する機能がある．具体的には，以下の機能がある．

- ワイルドカード検索
- ドメイン指定検索
- n-gram 検索
- 誤り候補の訂正支援機能
- 用例検索へのリンク

本稿では，改良を行う n-gram 検索のみ説明する[*1]．

3.3.1 *n*-gram 検索

第二言語で作文する際，作成した文中の誤りの有無やその位置が気になる．しかし，辞書を用いても誤りの有無やその位置を特定することは困難である．誤りの有無に対して，Web 上のヒット数を用いて判断する方法が考えられる．作成した文をフレーズ検索してヒット数を取得し，その値が低い場合は誤っている可能性が高いと判断する．しかし，一般的にフレーズが長くなるとヒット数は得られにくくなる．

そこで，先行ツールでは，入力文を分割して検索することで，文章中のどの箇所に誤りがあるのか，その候補を視覚的に表示する *n*-gram 検索機能を提案した．入力文を分割することで，ヒット数が得られやすくなり，誤りの有無が判定できるだけでなく，分割した系列まで誤り位置を絞り込むことが可能となる．分割方法は作文の対象言語によって異なり，英語などの場合は 1 単語（空白を利用），日本語の場合は文節（Yahoo! API を利用），韓国語の場合は文節相当（空白を利用），中国語の場合は 1 文字単位に分割する．分割後，*n*-gram 系列を生成し，各 *n*-gram 系列のヒット数を得た後，ヒット数に応じて 5 段階（ヒット数が高い方から，青・緑・黄・ピンク・赤）に色分けしたグラフでユーザに提示する．

例として，英語で “I wish you will be very happy together”（正解は，“hope you”）の誤り検証をした結果（2-gram～5-gram）を図 5 に示す．図 5 より，ヒット数が少なく，赤色で表示された “wish you will be very” に誤りを含む可能性があることがわかる．

しかし，*n* が大きくなる（フレーズが長くなる）とヒット数が得られず，“will be very happy together” のように正しいフレーズも誤りを含んでいると誤検出してしまう問題（問題点 A）がある．これは，訂正後の（正解）フレーズに対しても同様な問題が起こることを意味する．また，“wish you will be very” に誤り箇所を絞り込んでも，具体的な誤り位置はユーザ自身で絞り込む必要があるため，誤り位置を特定できない場合も想定できる（問題点 B）．したがって，以上の 2 つの問題点を改善する必要がある．

*1 他の機能については文献 [10),11)] を参照されたい．

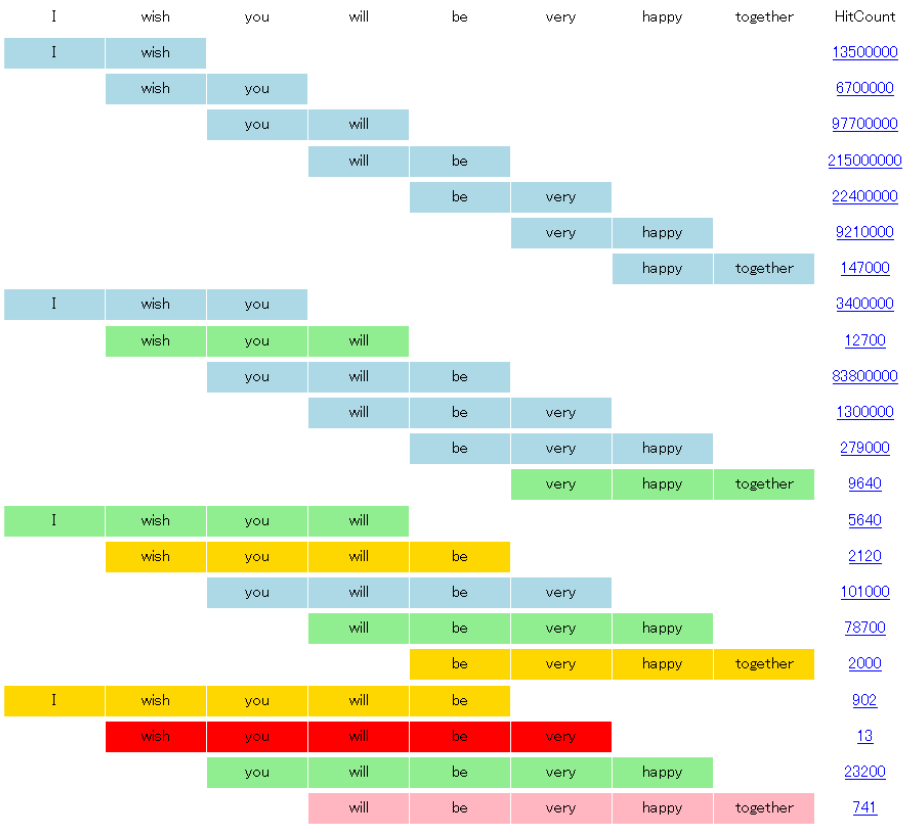

図 5　英語の *n*-gram 検索結果

3.3.2 n-gram 検索の改良

3.3.1 で述べたように，先行ツールの *n*-gram 検索では 2 つの問題がある．問題点 A に対しては，*n*-gram 系列の単独のヒット数以外の指標を用いてフレーズが長くなった場合にも対応できる手法が必要となる．問題点 B に対しては，誤り位置を絞り込む手法が必要となる．

そこで，問題点 A と B を改善する手法を考案するために，*n*-gram 検索結果を基に，ユーザが誤って書いた箇所の特徴分析を行った．誤りを含む英文 100 文 [12]に対して分析した結果，隣接する *n*-gram 系列のヒット数が急激に下がっている部分に誤りがある可能性が高いことがわかった．図 5 の 3-gram 系列 “I wish you” と “wish you will” のヒット数を比較すると，誤りを含む “wish you will” のヒット数が 2 桁も低くなっている．また，3-gram 系列 “wish you will” のヒット数と，“wish you will” を構成する 2-gram 系列 “wish you” と “you will” のヒット数を比較すると，3-gram 系列 “wish you will” のヒット数がかなり低い．これらの結果より，各 *n*-gram 系列のヒット数差と (*n*-1)-gram のヒット数差は，誤り判定の有用な情報になる可能性が高いと考えた．

以上の分析結果を基に，誤り候補検出を提案する．具体的には，各 *n*-gram において隣接する *m* 番目と *m*-1 番目の系列のヒット数差を示す $ScoreA_{nm}$ と，*n*-gram と (*n*-1)-gram のヒット数差を示す $ScoreB_{nm}$ を用いてスコア計算を行い，スコアが高い箇所を誤り候補として指摘する．また，これらのスコアを用いて，2-gram 上で誤り候補を指摘する絞り込み機能を実装した．提案手法により，2-gram を見れば，誤りの有無と位置を確認できるようになる．

提案する誤り検出手法の手順を以下に示す．

[検出フェーズ]

① 入力文を 1~5-gram 系列に分割し，*n*-gram 系列の *m* 番目のヒット数である H_{nm}（$n = 1 \sim 5$，$m = 1 \sim l_n$，l_n は *n*-gram 系列の個数）を Yahoo! API で取得する．

② 各 2～5-gram において，隣接する *m* 番目と *m*-1 番目（$m = 2 \sim l_n$，l_n は *n*-gram 系列の個数）の系列のヒット数差 $ScoreA_{nm}$（n=2～5）を以下の式で計算する．

$$ScoreA_{nm} = \log_{10}(H_{n(m-1)}) - \log_{10}(H_{nm}) = \log_{10}(H_{n(m-1)} / H_{nm})$$

③ ②で求めた $ScoreA_{n2}$ が負の場合（H_{n1} が H_{n2} より低い場合），$ScoreA_{n2}$ の絶対値を $ScoreA_{n1}$ とし，$ScoreA_{n2}$ の値は使用しない．先頭のフレーズのヒット数が低い場合，先頭のフレーズを誤り候補とするための処置である．

④ *n*-gram の *m* 番目の系列と (*n*-1)-gram の *m* および *m*+1 番目の系列のヒット数差 $ScoreB_{nm}$（n=2～5，m=1～l_n，l_n は *n*-gram 系列の個数）を以下の式で計算する．

$$ScoreB_{nm} = Min(\ ScoreN1\ ,\ ScoreN2\ )$$

$$ScoreN1 = \log_{10}(H_{(n-1)m}) - \log_{10}(H_{nm}) = \log_{10}(H_{(n-1)m} / H_{nm})$$

$$ScoreN2 = \log_{10}(H_{(n-1)(m+1)}) - \log_{10}(H_{nm}) = \log_{10}(H_{(n-1)(m+1)} / H_{nm})$$

ここで，*n*-gram の *m* 番目の系列は，(*n*-1)-gram の *m* および *m*+1 番目の系列を含むフレーズである．*n*-gram は，系列長が (*n*-1)-gram より長くなるため，(*n*-1)-gram の *m* および *m*+1 番目の系列のヒット数より，*n*-gram の *m* 番目の系列の方が低くなる．そのため，$ScoreB_{nm}$ の計算には，(*n*-1)-gram の最小ヒット数と *n*-gram 系列のヒット数との差，すなわち $ScoreN1$ と $ScoreN2$ の最小値を用いる．

⑤ ②と③で求めた $ScoreA_{nm}$，④で求めた $ScoreB_{nm}$ を用いて，各 *n*-gram の *m* 番目の系列の誤り候補判定値 $ScoreX_{nm}$（n=2～5，m=1～l_n，l_n は *n*-gram 系列の個数）を以下の式で計算する．

$$ScoreX_{nm} = ScoreA_{nm} + w \times ScoreB_{nm},\quad w:\ 重み$$

ここで, $ScoreA_{nm}$ は, 同一 n-gram において隣接する系列のヒット数差であるため, 誤りの無い系列ではヒット数差があまり生じない．それに対し，$ScoreB_{nm}$ は，n-gram 系列と(n-1)-gram 系列のヒット数差であるため, フレーズが長い n-gram 系列のヒット数の方が低くなり，必ずヒット数差が生じてしまう．そこで，重み w をかけて調整する．なお，予備実験により重み w を 0.6 とした．

⑥ 各 2～5-gram において, 誤り候補判定値 $ScoreX_{nm}$（n=2～5, m=1～l_n, l_n は, n-gram 系列の個数）が最大値の系列をそれぞれ抽出し，その値が閾値 $T_n(n = 2\sim5)$以上の系列を誤り候補として選出する．

[絞り込みフェーズ]

⑦ 選出した n-gram の誤り候補に対し，(n-1)-gram の m 番目のスコア $ScoreX_{(n-1)m}$ と m+1 番目のスコア $ScoreX_{(n-1)(m+1)}$のうち，高い系列に絞り込む．

⑧ ⑦を繰り返して，2-gram になるまで絞り込む．

⑨ ⑧で得た 2-gram を誤り候補としてグラフ表示する．

例として，“I want to buy car”（正解は “buy a car”）の誤り候補を検出する場合を考える．手順①により，1~5-gram のヒット数 H_{nm}（n=1～5，m=1～l_n，l_n は n-gram 系列の個数）を Yahoo! API で取得する. 例えば, 2~4-gram のヒット数は表 1 のようになる．

表 1 “I want to buy car” を構成する 2~4-gram のヒット数

H_{nm}	2-gram	ヒット数	H_{nm}	3-gram	ヒット数	H_{nm}	4-gram	ヒット数
H_{21}	I want	46100000	H_{31}	I want to	46100000	H_{41}	I want to buy	409000
H_{22}	want to	159000000	H_{32}	want to buy	16800000	H_{42}	want to buy car	1460
H_{23}	to buy	38600000	H_{33}	to buy car	18800			
H_{24}	buy car	144000						

手順②では，手順①で得たヒット数 H_{nm} を利用し，各 n-gram において隣接する m 番目と m-1 番目（m=2～l_n, l_n は n-gram 系列の個数）の系列のヒット数差 $ScoreA_{nm}$（n=2～5）を以下のように計算する．

計算例：2-gram の 2 番目 “want to” のスコア計算

$ScoreA_{22}$ = $\log_{10}$(“I want” のヒット数／“want to” のヒット数)

= $\log_{10}(H_{21}/H_{22})$ = $\log_{10}(46100000/159000000)$ = －0.5377

ここで，$ScoreA_{22}$ は負の値であるため，手順③の処理を行い，以下のように $ScoreA_{21}$ を算出し，$ScoreA_{22}$ は使用しない．

計算例：

$ScoreA_{21}$ = $Abs(ScoreA_{22})$ = 0.5377

以上の計算より，$ScoreA_{nm}$ の値は最終的に表 2 に示すようになる．

表 2 “I want to buy car” の各 n-gram 系列に対する $ScoreA_{nm}$ の計算結果

	$ScoreA_{nm}$		$ScoreA_{nm}$		$ScoreA_{nm}$
$ScoreA_{21}$	0.5377	$ScoreA_{31}$		$ScoreA_{41}$	
$ScoreA_{22}$		$ScoreA_{32}$	0.4384	$ScoreA_{42}$	2.4474
$ScoreA_{23}$	0.6148	$ScoreA_{33}$	2.9512		
$ScoreA_{24}$	2.4282				

手順④では，n-gram の m 番目の系列と(n-1)-gram の m および m+1 番目の系列のヒット数差 $ScoreB_{nm}$（n=2～5，m=1～l_n，l_n は n-gram 系列の個数）を算出する．

計算例：3-gram の 1 番目 “I want to” のスコア計算

$ScoreN1$ = $\log_{10}$(“I want” のヒット数／“I want to” のヒット数)

= $\log_{10}(H_{21}/H_{31})$ = $\log_{10}(46100000/4610000)$ = 0

$ScoreN2$ = $\log_{10}$(“want to” のヒット数／“I want to” のヒット数)

= $\log_{10}(H_{22}/H_{31})$ = $\log_{10}(159000000/4610000)$ = 0.5368

$ScoreB_{31}$ = Min(0 , 0.5368) = 0

以上の計算より，$ScoreB_{nm}$ の値は表 3 に示すようになる．

表 3 “I want to buy car” の各 n-gram 系列に対する $ScoreB_{nm}$ の計算結果

	$ScoreB_{nm}$		$ScoreB_{nm}$		$ScoreB_{nm}$
$ScoreB_{21}$	1.3697	$ScoreB_{31}$	0.0000	$ScoreB_{41}$	1.6136
$ScoreB_{22}$	0.8320	$ScoreB_{32}$	0.3613	$ScoreB_{42}$	1.1098
$ScoreB_{23}$	1.3240	$ScoreB_{33}$	0.8842		
$ScoreB_{24}$	3.4469				

手順⑤では，表 2（各 n-gram において，隣接する m 番目と m-1 番目の系列のヒット数差 $ScoreA_{nm}$（n=2～5，m=2～l_n，l_n は n-gram 系列の個数)），表 3（n-gram と(n-1)-gram のヒット数差 $ScoreB_{nm}$（n=2～5，m=2～l_n，l_n は n-gram 系列の個数)）の値を用いて，誤り候補判定値 $ScoreX_{nm}$ を算出する．

計算例：4-gram の 2 番目 “want tobuy car” のスコア計算

$ScoreX_{42}$ = $ScoreA_{42}$ + 0.6×$ScoreB_{42}$ = 2.4474+0.6×1.1098 = 3.1133

以上の計算より，$ScoreX_{nm}$ の値は表 4 に示すようになる．

表 4 “I want to buy car” の各 *n*-gram 系列に対する $ScoreX_{nm}$ の計算結果

	$ScoreX_{nm}$		$ScoreX_{nm}$		$ScoreX_{nm}$
$ScoreX_{21}$	1.3595	$ScoreX_{31}$		$ScoreX_{41}$	
$ScoreX_{22}$		$ScoreX_{32}$	0.6561	$ScoreX_{42}$	3.1133
$ScoreX_{23}$	1.4092	$ScoreX_{33}$	3.4817		
$ScoreX_{24}$	4.4963				

手順⑥では, *n*-gram 系列ごとに $ScoreX_{nm}$ が最大値の系列を選出する(表 4 の下線部). 選出した *n*-gram 系列の内, 表 5 の閾値 T_n 以上の系列を判定すると, $ScoreX_{42} > 3$ より, 4-gram の 2 番目の系列 “want to buy car” となるため, この系列を誤り候補として選出する. なお, 表 5 の閾値 T_n は, 予備実験により設定した値である.

表 5 各 *n*-gram の閾値 T_n

2-gram	3-gram	4-gram	5-gram
4.5	4	3	3

以上の計算により, 誤り候補 “want to buy car” を得る.

手順⑦からは絞り込みフェーズとなる. 手順⑥で得た誤り候補 “want to buy car” に対して, 誤り位置を絞り込む. 絞り込みには, 表 4 に示した誤り候補判定値 $ScoreX_{nm}$ を用いる. 手順⑥で得た誤り候補は, 4-gram の 2 番目の系列であるため, まず, (*n*-1)-gram である 3-gram の 2 番目 “want to buy” と 3 番目 “to buy car” の $ScoreX_{nm}$ を比較する. この場合, $ScoreX_{32}$(=0.6561) < $ScoreX_{33}$(=3.4817)より, スコアが高い 3-gram の 3 番目 “to buy car” に絞り込む.

次に, 手順⑧では, 絞り込んだ 3-gram の 3 番目の系列 “to buy car” に対し, (*n*-1)-gram である 2-gram の 3 番目 “to buy” と 4 番目 “buy car” の $ScoreX_{nm}$ を比較する. この場合, $ScoreX_{23}$(=0.6561) < $ScoreX_{24}$(=4.4963)より, スコアが高い 2-gram の 4 番目 “buy car” に絞り込む.

最後に, 手順⑨で, 絞り込んだ “buy car” を最終的な誤り候補とし, 図 6 のように 2-gram 上で赤色に表示する.

提案手法では, 最も誤りである可能性の高いものを選出するため, 誤りあり／なしの 2 種類で表示することができるため, 訂正後の妥当性も判断しやすい. なお, ユーザは, 絞り込み前の *n*-gram 検索結果も参照することができる.

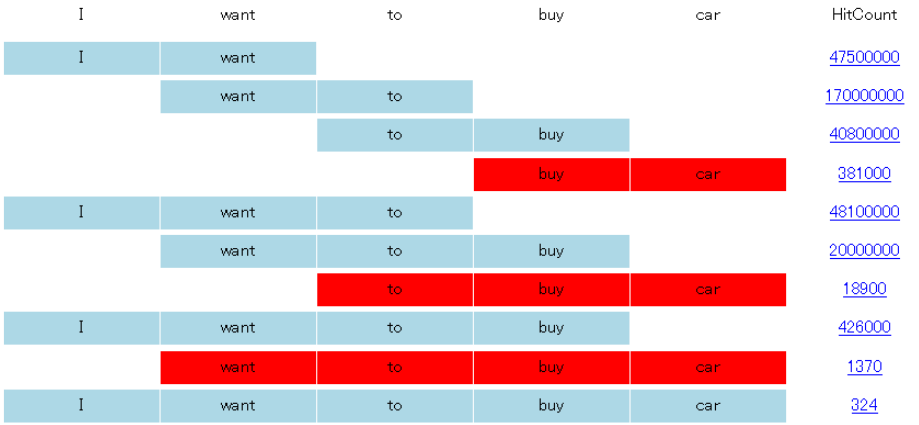

図 6 “I want to buy car”の *n*-gram 検索結果

4. 評価

先行ツールの誤り検出手法（3.3.1）と提案手法（3.3.2）の性能比較と提案手法の誤り候補に対する絞り込み手法の妥当性について評価する. 評価には, 誤りを含む英文と誤りを訂正した英文それぞれ 67 文 [13]を用いる. 正解データは, 67 文から誤り箇所を含む 2-gram を手作業で抽出した 67 個を利用し, 検出した系列中に正解データを含む割合（精度）で評価する.

4.1 誤り検出手法の誤り検出率

提案手法は, 誤り候補を検出（検出フェーズ）した後, 誤り位置の絞り込み（絞り込みフェーズ）を行うが, 先行手法では誤り候補の検出しか行わない. そこで, 提案手法の検出フェーズ後の結果と, 先行手法の検出結果を比較する. ここで, 提案手法は *n*-gram の範囲が $n = 2\sim5$ で固定であるため, 先行手法も *n* の範囲を 2~5 に設定する. また, 先行手法では *n*-gram 系列のヒット数に応じて 5 段階に色分けして表示するが, その中の 2 段階（ピンク色と赤色（800 件未満））までの系列を誤りと判断した.

比較結果を表 6 に示す. 表 6 より, 精度は, 提案手法の方が先行手法より 0.06 高いが, 両手法とも約 0.8 と高い精度を示している. また, 先行手法の総検出数は, 提案手法の約 3.4 倍も多いことが確認できる.

先行手法では, *n*-gram 系列のヒット数が 800 件未満（ピンク色と赤色）という固定

の値で誤りを判定しているため，*n*-gram の *n* の値が大きくなる（系列のフレーズが長くなる）とヒット数が得られにくくなり，誤り候補の総検出数が増加する．しかし，総検出数が増加しても，フレーズが長いとフレーズ中に正解データが含まれる可能性が高くなる．また，先行手法では，ヒット数に応じて 5 段階に色分けして表示しているため，ユーザはその色を確認して独自に誤りを判定する必要がある．そのため，ユーザごとに誤りを判断する基準が変わってしまう可能性がある．今回は，ヒット数 800 件未満（2 段階）までを誤りと判定したが，ヒット数 20 件未満（1 段階）やヒット数 5000 件未満（3 段階）までを誤り検出の基準とすれば，検出結果も異なる．

一方，ヒット数差を利用した提案手法では，色分けの指標が誤りあり／なしの 2 種類なので判定が容易である．また，*n*-gram 系列のヒット数差を用いて誤りを検出しているため，フレーズが長くなりヒット数が得られにくくなっても，ヒット数差が小さければ誤りとして検出せず，総検出数が少なくなる．しかし，隣接する *n*-gram 系列のヒット数が急激に下がっている部分は誤っている可能性が高いため，総検出数が少なくなっても高い精度で誤りを検出できる．さらに，言語によって得られるヒット数の桁数が増減しても，ヒット数差が大きい部分を誤りとして検出できるため，言語に対する依存性もない．これらの結果より，誤り検出については，提案手法の方が有用であるといえる．

表 6　誤り候補の検出精度

誤り検出手法	先行手法	提案手法
総検出数	303	90
正解数	236	76
精度	0.78	0.84

4.2 誤り訂正文（妥当な文）に対する誤検出

ツールで誤りを検出した後，ユーザは入力文を訂正することになるが，訂正後の妥当性判定もツールで行うことになる．その際，誤り訂正文（妥当な文）に対して誤検出してしまうと，訂正できたかどうかの判断ができない．そこで，訂正した英文（妥当な文）67 文を利用して，両手法の誤検出精度について調査する．

調査結果を表 7 に示す．表 7 より，提案手法の誤検出数は先行手法より，約 82.2%も減少していることがわかる．先行手法は，誤りの検出に *n*-gram 系列のヒット数しか利用していないため，妥当な文でも *n* が大きくなるとヒット数が得られにくくなるため誤検出が増加すると考えられる．また，表 6 と表 7 の総検出数を比較すると，誤りを正しく検出した時と誤検出した時で総検出数が同程度となっている．これらの結果から，先行手法では誤り訂正文（妥当な文）に対しても，誤検出する可能性が高いといえる．また，ヒット数に応じて 5 段階に色分けして表示しているため，誤りを検出した色からどの色になれば訂正できたかも判断しづらい．

一方，提案手法の誤検出については，訂正後のフレーズのヒット数差が極端に大きい場合に誤検出していることがわかった．例えば，“I want to go abroad next year” において，“to go” のヒット数が 71,200,000 件に対して “go abroad” のヒット数は 94,000 件であった．そのため，妥当な文であるにもかかわらずヒット数差が極端に大きくなってしまい，“go abroad” を誤りとして誤検出した．しかし，提案手法では，誤りを示す色分け指標が誤りあり／なしの 2 種類であるため，訂正後の妥当性を確認しやすい．また，先行手法より誤検出数が非常に少ないため提案手法の方が有用であるといえる．

表 7　先行手法と提案手法の誤検出比較

誤り検出手法	先行手法	提案手法
総検出数	202	36

4.3 品詞別の誤り検出率

品詞別の誤り検出の傾向を調査するため，表 6 の結果に対し，先行手法と提案手法で検出した誤りを品詞別に分類整理する．

分類結果を表 8 と表 9 に示す．なお，表中の「その他」は複数形など品詞以外のデータをまとめたものである．表 8 と表 9 の各品詞の精度より，動詞・形容詞については先行手法の方が平均で 0.11 程度高く，冠詞については提案手法の方が 0.39 高くなった．形容詞誤りに対して，提案手法は先行手法より 0.18 低い値となった．提案手法と先行手法での形容詞誤りに対する検出ミスは 2 件であるが，提案手法は検出数が少ないため精度が低い結果となった．また提案手法では，すべての品詞において，精度が 0.71 以上あるため，品詞に依存することなく，誤りを検出できるといえる．一方，先行手法では，冠詞誤りの精度が 0.54 と低く，冠詞誤りに弱いと言える．ただし，検証データ数が少ないため，大規模データでの検証が必要である．

表 8　先行手法の品詞別誤り検出

誤り検出手法	冠詞	動詞	形容詞	副詞	名詞	前置詞	その他
検証文の総数	12	11	6	4	5	10	19
総検出数	59	58	18	13	24	53	78
正解数	32	48	16	13	23	52	52
精度	0.54	0.83	0.89	1.00	0.96	0.98	0.67

表 9　先行手法の品詞別誤り検出

誤り検出手法	冠詞	動詞	形容詞	副詞	名詞	前置詞	その他
検証文の総数	12	11	6	4	5	10	19
総検出数	14	14	7	7	5	14	29
正解数	13	11	5	7	5	14	21
精度	0.93	0.79	0.71	1.00	1.00	1.00	0.72

4.4 提案手法の絞り込み精度

提案手法では，誤り候補を検出した後，*n*-gram 系列のヒット数差で算出したスコア $ScoreX_{nm}$ を用いて，2-gram 上で誤りを指摘する絞り込み機能を実装している．そこで，絞り込み機能の有用性を評価するために，提案手法で検出した *n*-gram 系列に対し，絞り込みの精度を調査する．

調査結果を表 10 に示す．表 10 より，絞り込みにより，精度が 0.20 低下していることが確認できる．精度が低下する原因を調査した結果，隣接する系列のヒット数差 $ScoreA_{nm}$ の計算において *n*-gram 系列のヒット数差を用いるが，正解データの系列（誤りを含む系列）より，それ以外の系列のヒット数差が大きくなることで，正解データの系列以外のスコアが高くなる状況が確認できた．例えば，“There is someone at front door” から検出した誤り候補 “someone at front” に対して，“someone at”（スコア 2.636）と “at front”（スコア 2.072）のどちらかに誤り系列を絞り込む場合，誤りではないがスコアの高い “someone at” に絞り込まれてしまう．この原因を調査したところ，“someone at” のスコアは，“is someone”（ヒット数 1,270,000 件）と “someone at”（ヒット数 246,000 件）のヒット数差で算出されるが，この系列間のヒット数差が大きい．一方，“at front” のスコアは，“someone at”（ヒット数 246,000 件）と “at front”（ヒット数 200,000 件）のヒット数差で算出されるが，こちらの系列間はヒット数差が小さい．その結果，ヒット数差が大きい “someone at” のスコアが高くなってしまう．Web 上には誤りではないが，あまり利用されない表現が存在する．これが影響した例であるといえる．

以上より，誤りの絞り込み精度については，もう少し改善する必要がある．ただし，現状でも，絞り込み過程である 3~5-gram の結果を参照できるため，ユーザはツールの判定を検証することは可能である．

表 10　提案手法の絞り込み調査

	絞り込み前	絞り込み後
総検出数	90	61
正解数	76	39
精度	0.84	0.64

5. おわりに

本稿では，9 カ国の言語に対する作文支援ツールの改良として，*n*-gram 系列のヒット数差を用いた誤り候補検出手法を提案した．*n*-gram 系列のヒット数差を用いて誤り候補を検出し，2-gram まで誤り箇所を絞り込むことができる．また，誤りあり／なしの 2 種類で提示するため，訂正できたかどうかを確認しやすい．

誤り検出手法の性能を評価したところ，誤り検出については精度 0.84 と高い値を示した．しかし，誤り位置を 2-gram まで絞り込んだ際の精度は 0.64 となり，絞り込み前より精度が 0.20 低下した．今後の課題として，検出した誤りに対して，精度の高い絞り込み方法の検討が挙げられる．さらに，多言語の大規模データで検証評価を行う必要がある．そして，より使いやすい作文支援ツールの構築を目指す．

謝辞　本研究は，文部科学省科学研究費補助金（若手研究(B) 19700638）の一部の補助を受けて実施した．

参考文献

1)　大鹿広憲，佐藤学，安藤進，山名早人: Google を活用した英作文支援システムの構築,DEWS2005, 4B-i8 (2005).
2)　綱嶋祐一，岡田壮史，安藤一秋: 検索エンジンを利用した多言語作文支援，電子情報通信学会技術研究報告，ET2007-97, Vol.107, No.536, pp.73-78 (2008).
3)　佐藤学，安藤進，山名早人: 検索エンジンを利用した英作文支援システムの構築，言語処理学会第 12 回年次大会論文集, pp.664-667 (2006).
4)　平野孝佳，平手勇宇，山名早人: 検索エンジンを用いた英文冠詞誤りの検出，電子情報通信学会技術研究報告，DE2007-48, Vol.107, No.131, pp.157-162 (2007).
5)　久保田朗，太田学: 検索エンジンを用いた英文前置詞誤り修正のための検索フレーズ生成法，情報処理学会研究報告, Vol.2010-DBS-151, No.37 (2010).
6)　谷本太郁由，太田学: 検索エンジンを用いた動詞名詞コロケーションに基づく英文動詞誤りの検出と修正，情報処理学会研究報告, 2010-DBS-151 (2010).
7)　英文校正サイト [NativeChecker], http://native-checker.com/
8)　統計的日本語校正デモ，http://cl.naist.jp/chantokun/
9)　Yahoo!デベロッパーネットワーク, http://developer.yahoo.co.jp/
10)　古川陽平，綱島祐一，安藤一秋: Web 検索を利用した 9 ヶ国語作文支援の改良，電子情報通信学会技術研究報告，ET2009-125, Vol.109, No.453, pp.125-130 (2010).
11)　古川陽平，綱島祐一，安藤一秋: 検索エンジンを利用した 9 ヶ国語作文支援の拡張，第 9 回情報科学技術フォーラム（FIT2010）講演論文集, pp.551-552 (2010).
12)　よくある英語のミス: 目次 - K-TAN's Website, http://k-tan.staba.jp/home/errors/index.html
13)　ジェイムズ・H・M・ウェブ: 日本人に共通する英語のミス１２１ 改訂新版，ジャパンタイムズ (1991).