KWISC —文脈理解のための係り受け表示手法—

荒井 裕介，小川 泰弘，外山 勝彦（名古屋大学）

1　はじめに

用例文を参照しながら文を作成するとき，用例文に出現する語の意味だけでなく，その語の文脈を理解することは重要である．文脈は，語の近接関係や依存関係などから決まると考えられ，それらの関係を把握することは文脈の理解につながる．

語の文脈を理解するための手法の一つとして，KWIC(KeyWord In Context)[1] がある．KWIC は，文中に出現するキーワードを中央に揃えて，各文を表示する手法である．キーワードの前後の文字列をキーとしてソートすることにより，キーワードと他の語との共起関係（コロケーション）の発見を容易にする．しかし，KWIC は，以下の点で不十分である．

第 1 に，KWIC は，語の近接関係を表示できるが，依存関係を表示できない．キーワード周辺に出現する語の中には，キーワードとの依存関係がない語も存在する．図 1 に,「犬–飼う」というコロケーションを含む文を KWIC により表示した例を示す．1 行目において，キーワード「犬」の周辺に「買う」が出現しているため，それらの間に依存関係があるという誤解を招きやすく,「犬–飼う」というコロケーションを見逃す可能性がある．

第 2 に，KWIC では，キーワード周辺以外の部分に出現する語とのコロケーションの発見が容易でない．図 1 において，共起する 2 語の間に挿入句や修飾語があり，「飼う」がキーワード周辺や，キーワードから離れた位置など様々な位置に表示されている．そのために,「犬–飼う」というコロケーションの存在を見逃す可能性がある．また，一画面に収まらないほど離れた位置に出現する語とのコロケーションの発見は，さらに容易でない．

これらの問題点に対して，秋山らは，TextImi[3][4] を提案した．図 1 と同じ文を TextImi により表示した場合の例を図 2 に示す．TextImi は，日本語文から単文を抽出し，そこから文の骨格（文末の文節とそれに係る文節）のみを表示するツールである．それらの文節は,「は」「が」「を」「に」「で」「その他」「述部」の順序で固定された列に配置される．ここで,「述部」の列には，文末の文節が配置される．それに係る文節のうち，助詞「は」「が」「を」「に」「で」のいずれかを含む文節は，その助詞の列に配置され，残りの文節は，

太郎は **犬** を買ったばかりの小屋で飼う
太郎は **犬** を小屋で飼う
息子のために初めて **犬** を飼う

図 1: KWIC による表示例

「その他」の列に配置される．文の骨格となる文節のみを表示するため，挿入句や修飾語は除外され，KWIC では困難であったコロケーションの発見を容易にしている．

しかし，TextImi は，固定した列に文節を配置しているため，文の語順を変えて表示する場合があるという問題点がある．さらに，その場合には，共起する 2 語が元の文に比べて，より離れて表示されることがある．図 2 の 3 行目では,「ために」と「初めて」が,「犬を」を越えて後方へ移動し，そのため,「犬を」と「飼う」がより離れて表示されている．

本稿では，KWIC における以上 2 つの問題点を解決するために，TextImi とは異なるアプローチをとり，文脈理解のための係り受け表示手法 KWISC(KeyWord In Structured Context) を提案する．

2　提案手法

本節では，日本語文を対象として，KWIC における 2 つの問題点を解決するための手法 KWISC について述べる．

2.1　係り受け関係の表示

第 1 の問題点を解決するために，KWISC では，係り受け関係を表示した．TextImi と同様に，文を文節単位で区切り，文末の文節とそれに係る文節を文の骨格として扱う．異なる点は，表示する文節を文の骨格に限らない点と，キーワードを含む文節を揃え，前後の文節をその左右に配置する点である．係り受け関係を表示した例を図 3 に示す．各文節は，文末の文節からの係り受けの深さにしたがって階層的に表示される．すなわち，係り受けが深くなるほど上に表示される．それにより，文節間の係り受け関係を把握することができる．

2.2　第 2 キーワードの指定

第 2 の問題点を解決するために，KWISC では，もう 1 つのキーワード（第 2 キーワード）を指定可能に

は	が	を	に	で	その他	述部	文
太郎は		犬を		小屋で		飼う	太郎は犬を買ったばかりの小屋で飼う
太郎は		犬を		小屋で		飼う	太郎は犬を小屋で飼う
		犬を	ために		初めて	飼う	息子のために初めて犬を飼う

図 2: TextImi による表示例

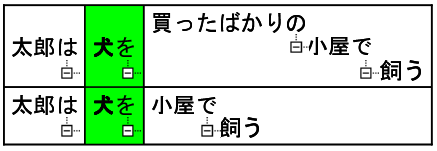

図 3: 係り受け関係を表示した例

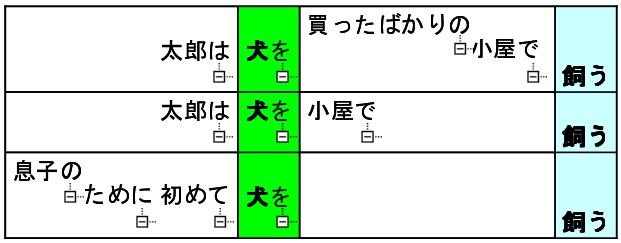

図 4: 第 2 キーワードを指定した例

した．まず，1 つのキーワード（第 1 キーワード）により検索する．その検索結果に対して，第 1 キーワード以外の部分において，第 2 キーワードを指定することにより，各キーワードを含む文節をキーワードごとに揃えて配置する．図 4 に，第 1 キーワードが「犬」，第 2 キーワードが「飼う」である例を示す．様々な第 2 キーワードを指定することにより，キーワード周辺に出現する語であるかどうかに関わらず，コロケーションを発見することが可能になる．

また，第 2 キーワードの前後の文字列をキーとしてソートすることにより，3 語からなるコロケーションの発見も可能になる．

TextImi と異なる点は，語順を変えない点と，助詞「は」「が」「を」「に」「で」に限らず，任意の語により文節を揃えて配置できる点である．語順を変えないため，他の語が移動してきて，共起する 2 語がより離れることはない．また，第 2 キーワードには名詞や形容詞も指定できるため，それらとのコロケーションを発見することもできる．

2.3 文節の展開/折りたたみ

2.2 節において述べた手法により，共起する 2 語を含む文節をそれぞれ揃えて表示することができる．しかし，その際には，第 1 キーワードから最も離れている位置で揃うために，共起する 2 語がより離れて表示される文も存在する．

そこで，KWISC では，階層的に表示されている文節を，フォルダツリーのように展開/折りたたみ可能にした．各文節の表示/非表示を段階的に選択して，共起する 2 語の間にある挿入句や修飾語を非表示にできる．文節を折りたたむ際には，それより係り受けの深さが深い文節も一緒に折りたたまれ，代わりに文節展開ボタンが表示される．図 4 に対して，キーワードを含む文節以外を折りたたんだ例を図 5 に示す．

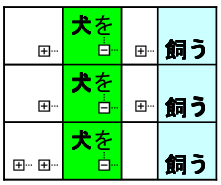

図 5: 文節を折りたたんだ例

また，KWISC における検索結果の初期表示では，TextImi と同様に文の骨格のみが表示され，残りの文節は非表示である．例外として，キーワードを含む文節が文の骨格に含まれていない場合には，そこから文の骨格までの文節をすべて表示する．

3 評価実験

KWISC は，文の骨格のみを表示し，さらに各文節を展開/折りたたみできるため，共起する 2 語の間を短い距離で表示できる．この距離の短さは，コロケーションの発見の容易さと関係があると考えられる．すなわち，距離が短いほど共起する 2 語が一画面に収まりやすく，さらに，コロケーションを探す際に左右への視線の移動が少なくて済むと考えられる．そこで，本節では，KWISC における 2 つの文節間の表示上の距離を測定し，関連研究との比較を行った．

本実験では，コーパスとして，EDR コーパス [5] 207,802 文を用いた．また，構文解析には CaboCha[2] を用いた．

なお，画面上における距離を実際に測定することは困難であるため，本実験では，2 つの文節間の文字数を表示上の距離として測定した．その際，画面上に表示されるフォントを等幅フォントとし，全角文字 1 文字分の幅を距離 2，半角文字 1 文字分の幅を距離 1 として測定した．また，KWISC における文節展開ボタンは，距離 2 として測定した．

3.1 KWIC との比較

第 1 の実験では，文をそのまま表示する KWIC と，文の骨格のみを表示する KWISC において，文節間距離を測定し，KWIC に対する KWISC の有効性を確認

表 1: 文節間距離の平均と標準偏差

	平均	標準偏差
KWIC	24.39	22.75
KWISC	9.90	9.78

する．

3.1.1 実験方法

本実験では，名詞を含む文節とそれが係る文節との間の距離を測定した．KWIC と KWISC を用いて，キーワードにより検索した結果から，そのキーワードと他の語とのコロケーションを発見する場合を想定した．

そこで，EDR コーパスから，名詞を含む文節と，それが係る文節からなる組を 745,837 組収集した．ただし，2 つの文節がすでに隣り合っている組は除いた．これは，これ以上距離を短くする必要がないためである．その結果得られた 321,286 組に対して，文節間距離を測定した．また，KWISC による文節間距離の短縮率も測定した．短縮率は，式 (1) により求めた．

$$\text{短縮率} = 1 \quad \left(\frac{\text{KWISC における文節間距離}}{\text{KWIC における文節間距離}} \right) \quad (1)$$

3.1.2 結果と考察

文節間距離の平均と標準偏差を表 1 に示す．いずれも KWISC では，KWIC の半分以下となっている．このことから，KWISC によって，コロケーションの発見が容易になるといえる．

次に，KWISC による文節間距離の短縮率の分布を図 6 に示す．共起する 2 語の間にあるすべての文節が，文の骨格に含まれている場合，短縮率は 0 となる．そのような組を除いた 253,266 組 (78.8%) については，文節間距離を短縮することができたため，KWISC の効果があるといえる．また，短縮率が 0 となる組も含めた平均短縮率は 0.51 であるが，短縮率は，0.6〜0.9 の間に多く集まっている．ここから，KWISC は，共起する 2 語の間に挿入句や修飾語を持つ文に対して，その間の距離を半分以上短縮することができるといえる．

次に，KWIC における文節間距離と KWISC による短縮率の関係を図 7 に示す．KWIC における文節間距離が 20 以下の場合は，短縮率が 0.5 以下であるが，その距離が長くなるにしたがって短縮率が上昇する．ここから，KWIC における文節間距離が長いほど，KWISC による短縮率が高く，有効であると確認できた．

3.2 TextImi との比較

第 2 の実験では，語順を変える場合のある TextImi と，語順を変えない KWISC において，文節間距離を測定し，TextImi に対する KWISC の有効性を確認する．

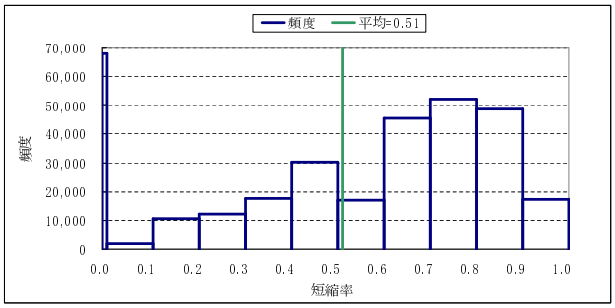

図 6: 文節間距離の短縮率の分布

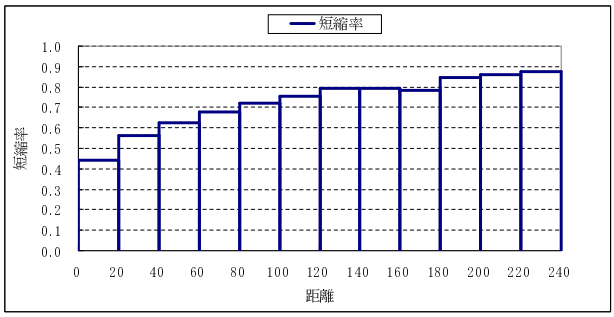

図 7: 文節間距離と平均短縮率の関係

3.2.1 実験方法

本実験では，助詞「は」「が」「を」「に」「で」のいずれかを含む文節と，それが係る文節との間の距離を測定した．TextImi においては，述部により検索した場合を想定し，一方，KWISC においては，述部により検索した結果から第 2 キーワードとして助詞「は」「が」「を」「に」「で」のいずれかを指定した場合を想定した．

そこで，EDR コーパスから，助詞「は」「が」「を」「に」「で」のいずれかを含む文節と，それが係る文節（述部）からなる組を 511,982 組収集した．ただし，一般的には，コロケーションは複数の文を見比べて発見するものであるため，出現回数が 1 回である組は除いた．その結果得られた 500,483 組に対して，8,677 種類の述部ごとに文節間の距離を測定した．助詞ごとの内訳を表 2 に示す．

3.2.2 結果と考察

助詞ごとの平均文節間距離を表 3 に示す．助詞「で」を除いて，KWISC は，TextImi よりも文節間距離が短い結果となった．TextImi における距離が長かった原因は，語順を変えたためである．TextImi では，助詞「は」「が」「を」「に」「で」を含む文節以外のものを，すべて「その他」の列にまとめて配置している．そのため，他の語が共起する 2 語の間に移動し，その間の

表 2: 実験に用いる文節の組

	は	が	を	に	で
組の数	108,681	102,179	145,307	109,099	35,217
述部の種類	5,095	4,615	5,128	4,471	2,893

図 8: システムの動作例

表 3: 助詞ごとの平均文節間距離

	は	が	を	に	で
TextImi	45.1	32.2	27.9	19.1	14.5
KWISC	29.3	16.2	9.8	14.2	18.5

距離が長くなる．

次に，助詞「で」について，TextImi よりも KWISC の方が距離が長かった．この原因は，助詞「で」が，助詞「は」「が」「を」「に」よりも前方に出現する文があったためである．それに対して，助詞が「は」「が」「を」「に」「で」の順序にしたがって出現する文の割合は，90.1%であった．このことから，KWISC が TextImi よりも距離が長くなる可能性は低いといえる．

次に，助詞「は」について，KWISC における平均距離は，TextImi よりも短かったとはいえ，29.3 と他の助詞よりも長くなった．この原因は，キーワードを含む文節を揃えているためである．第 2 キーワードを指定した際には，第 1 キーワードとの間の距離が最大である文と同じ距離に揃えられてしまう．その場合にも，KWISC では，各文節を折りたたむことによって距離を短くできるため，コロケーションの発見は容易である．

4 KWISC の実装

KWISC を Web アプリケーションとして実装した．クライアント側の実装には JavaScript を，サーバ側の実装には Ruby を用いた．本システムは，構文解析済みの日本語コーパスを検索対象とする．ただし，その係り受け関係は，(1) 文末の文節を除き，各文節はその後方に係り先を 1 つだけ持つ，(2) 係り受け関係は交差しない，という制約を満たすものとする．この制約を満たすコーパスとして，現在は，CaboCha[2] によって構文解析されたコーパスを用いている．システムの動作例を図 8 に示す．

入力する検索キーワードは，文節をまたがない文字列である．第 2 キーワードを指定するためには，文節の主辞部または機能部をクリックする．ここで，主辞部とは，文節内において品詞が助詞・接尾辞である形態素より前の部分をいい，機能部とは，それ以降の部分で句読点などの記号を除いたものをいう．これらにカーソルを合わせた場合，各行の第 2 キーワード候補は赤い表示に変わり，それを含む文節には下線が引かれる．各キーワードを含む列のヘッダには，そのキーワードが表示される．また，文節を展開したり，折りたたんだりするためには，各文節の前にあるボタンをクリックし，各列をソートするためには，その列のヘッダをクリックする．

5 おわりに

本稿では，KWIC における 2 つの問題点を解決するため，文脈理解のための係り受け表示手法 KWISC を提案した．KWISC は，係り受け関係の表示，第 2 キーワードの指定による文節の整列，文節を折りたたむことによる文節間距離の短縮が可能である．関連研究との比較実験により，共起する 2 語の間の距離が短縮できることを確認した．そのため，コロケーションの発見が容易になるといえる．

今後の課題として，日本語以外の言語への対応が挙げられる．そのためには，前節で述べた制約を超えて，前方要素への依存，複数要素への依存，依存関係の交差に対応する必要がある．

参考文献

[1] Luhn, H.P.: Key-Word-In-Context Index for Technical Literature(KWIC Index), IBM Corporation, Yorktown Heights, NY (1959).

[2] 工藤拓，松本裕治：チャンキングの段階適用による日本語係り受け解析，情報処理学会論文誌，Vol.43，No.6，pp.63–69 (2002).

[3] 秋山優，深谷昌弘，大岩元，舘野昌一：文構造の標準化による Kwic の拡張，情報処理学会研究報告 自然言語処理研究会報告，Vol.2006，No.94，pp.105–112 (2006).

[4] 中野智仁：大量テクストの意味分析を可能とする日本語テクスト解析ツール TextImi の開発，総合政策学ワーキングペーパーシリーズ，No.115 (2007).

[5] 日本電子化辞書研究所：EDR 電子化辞書日本語コーパス，(1995).