学習者コーパス入門

学習者コーパスのあるべき条件とその作成の具体的方法は?

英語学習者の発話や作文などの performance data を
コンピュータ処理し, 教育や研究に役立てるために

第4回 「コーパス・データの管理と検索の基礎」

元東京学芸大学講師／ランカスター大学言語学科博士課程在籍 投野 由紀夫

はじめに

前回は, 学習者データの transcription の実際的な方法と, コーパス・データの基本フォーマットに関して説明し, その一例として XML を簡単に紹介した。今回は前半で XML を採用する利点についてもう少し補足し, ファイル管理のポイントをまとめたうえで, 後半はいよいよ実際のコンコーダンサーをいじりながらコーパス検索の基本的な考え方を整理してみたい。

XML 採用の利点

XML は大規模コーパス・データの標準フォーマットとして BNC (British National Corpus) の2nd version や CES (Corpus Encoding Standard) などが採用している。その最大の理由は電子テキスト企画の今後の標準となる可能性がきわめて高いからである。SGMLと同様の柔軟でかつ詳細な文書構造が定義できるのと同時にインターネットに対応したハイパーテキストの機能を有している。今後, コーパスが音声や動画, あるいは文書に付随するイメージなどを補足情報として内蔵していく可能性や, コーパスデータから派生していろいろなソフトウェア (たとえばコンコーダンサー, リレーショナル・データベース, コースウェアなど) を開発する際の「データの意味を説明するメタ言語」として XML の存在が重要になってきているのだ。

XMLそのものの詳細な説明はスペースがないのでできないが, 学習者コーパス構築という観点で素人のわれわれが利用する際にもいくつか利点がある。

①文書作成のツール類が豊富に揃っている

まず XML というとむずかしそうに感じるが, 基本的な構造の定義の仕方さえ勉強すれば, あとは補助ツールがたくさんあるのでそれらを使いまわせる可能性が高い。たとえば,

図1

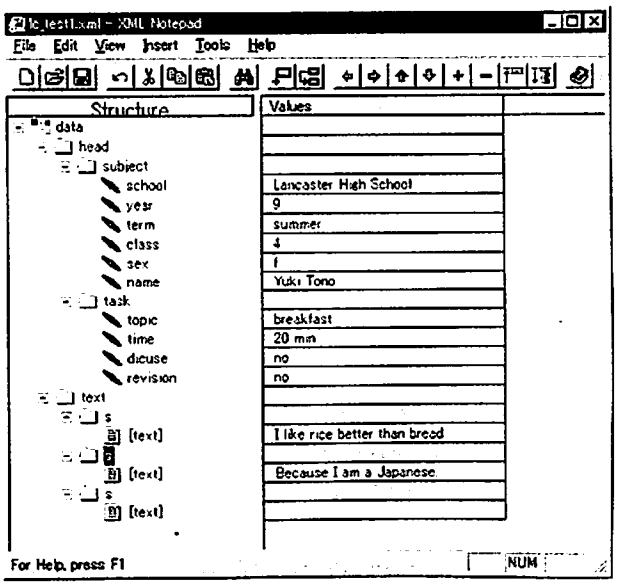


図2

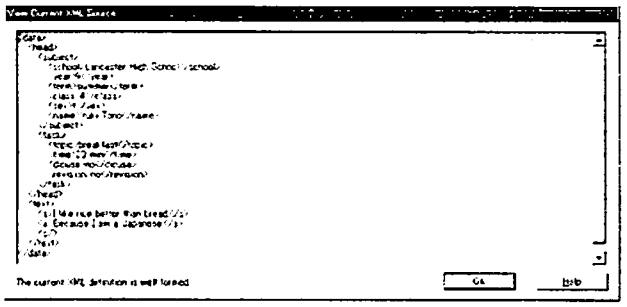


図1はマイクロソフトがベータ版で公開しているXML Notepadというxml専用エディタである(URL:http://msdn.microsoft.com/xml/notepad/)。

図1の左側のウィンドウのツリーの部分を自分で定義して、右側のValuesの部分に実際の値を入れていけば、XMLのタグ形式をほとんど意識せずに図2のような形式のwell-formed XML文書が作成できる。テキスト・ファイルでの操作になれている人には少々回りくどいインターフェースだが、タグ処理がよくわからない人にはこのエディタは便利だ。

もう少し高度な処理がしたい方には、XMLwriter(http://www.xmlwriter.net)がお勧めである。軽快な動作をするエディタ兼パーザーで、プロジェクト単位でXML文書を管理したりXMLとしての書式の検証などができる。ここですべての機能の紹介をすることはできないが、インターネット上でいろいろなツールがフリーや試用版で試せるので、興味のある方はxmlpitstop.com(http://www.xmlpitstop.com/)などに目を通してみることをお勧めする。

②コーパスの文書構造チェックが容易に可能

学習者データの量が増えてくると、マニュアルでタグを付けていたものに関してはどうしてもタグの挿入ミスや文書構造違反などが出てくる。この際にXMLは威力を発揮する。XMLではこの文書内の構造チェックをvalidationといってとくに重要視しており、そのためのツール類もたくさん出回っているので、これを利用可能だ。

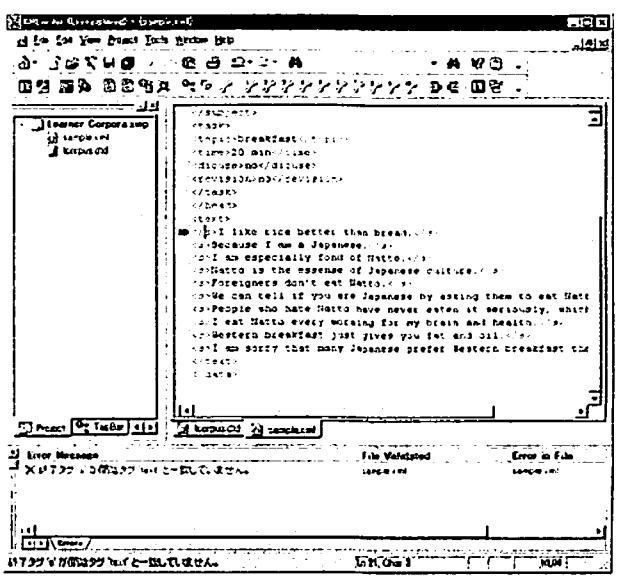
```
<!ELEMENT data (head, text)                              >
<!ELEMENT head (subject, task)                           >
<!ELEMENT
    subject(school,year,term?,class?sex?name?)>
<!ELEMENT task (topic, time, dicuse, revision)>
<!ELEMENT text (s*)                                      >
<!ELEMENT school (#PCDATA)                               >
<!ELEMENT year (#PCDATA)                                 >
<!ELEMENT term (#PCDATA)                                 >
<!ELEMENT class (#PCDATA)                                >
<!ELEMENT sex (#PCDATA)                                  >
<!ELEMENT name (#PCDATA)                                 >
<!ELEMENT topic (#PCDATA)                                >
<!ELEMENT time (#PCDATA)                                 >
<!ELEMENT dicuse (#PCDATA)                               >
<!ELEMENT revision (#PCDATA)                             >
<!ELEMENT s (#PCDATA)                                    >
```

XMLでは文書構造の定義はDTD(**文書型定義**)によって行う。DTDには単にタグ付けで使用する要素や属性が定義されるだけでなく、文書の1つ1つの部分の全体に対する関係などの「意味付け」が行われる。文書構造がしっかりしていれば、XML文書としての利用価値がそれだけ高くなる。

第3回でごく基本的な学習者コーパスのファイルのフォーマットを紹介してtest.xmlというファイルに保存したが、これに対応するDTDは左図のようになる。個々の変数の値を属性として細かく定義することも可能だが、ここではDTDのイメージがわかるように非常に簡略にしてある。これを外部サブセットとして1つのDTDファイルとして保存して、XMLフォーマットで書かれた学習者コーパス・データから参照させるように指定することもできるし、これを各ファイルの先頭に置いて、内部サブセットとして指定することも可能だ。

図3はこのDTDを用いて、XMLwriterで文書構造の検証をさせているところである。パーザーが構文チェックをして<text>中の、<s>の開始タグの誤りを指摘している。このようにDTDによってコーパスデータの持つさまざまな情報

図3



をきちんと定義してやることで，コーパスデータが「情報利用」という観点から意味を持ってくる。さらに，一般の XML ツールが自動的に大量のファイルの検証を行ってくれるので，特別に個別のプログラムなどを書かなくてもよい。こういった点で，XML 文書の形式でコーパスをフォーマットするのは今後の方向性として非常に重要なことなのである。

コーパス・ファイル管理のヒント

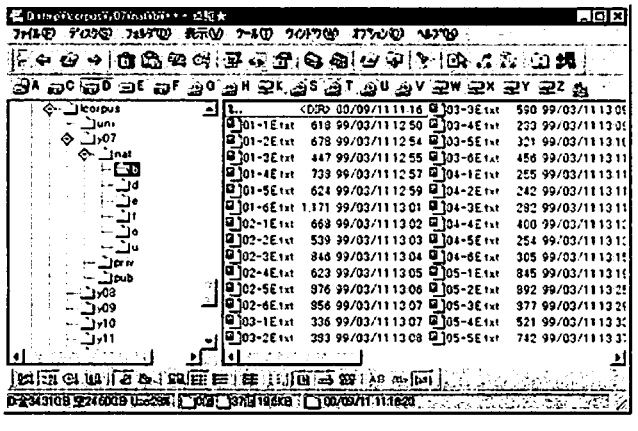
今回のように XML ですべてのファイルをフォーマットすることができればいいが，現実的にはなかなか時間的な制約があってそうはいかないかもしれない。そんなときにはあまりあせらずに段階的にファイル管理の仕組みを作っていくとよい。

ファイル管理の段階

①ディレクトリによるファイル整理
②ファイル名による管理
③ヘッダによる管理

まず第一段階は，とりあえず transcription だけは作っておいて，それをディレクトリによって管理するという方法だ。実はこれがもっとも一般的に行われている文書管理の方法かもしれない。図4はそのディレクトリ管理の一例である。lcorpus というディレクトリの下にy07（中学1年）からuni（大学）まで7グループのサブディレクトリを作り，各サブディレクトリの下にnat（国立付属），priv（私立），pub（公立）といった学校分類をして，またその下にエッセイの課題テーマ別のサブディレクトリを作っている。この方法だと比較的

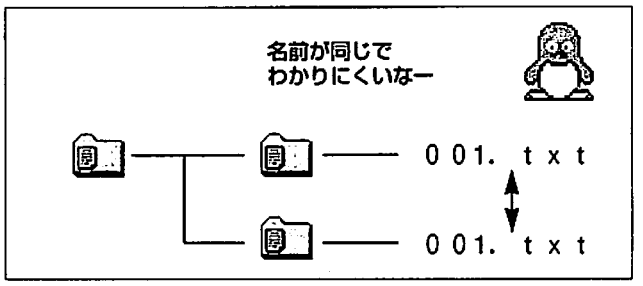
図4



簡単に分類が可能で，WordSmith や TXTANA などのコンコーダンサーにはフォルダごとコーパス登録しておく機能があるので，サブコーパスを指定するのも容易だ。

しかし，この方法ではサブコーパスとして比較する変数（学年，学校，トピックなど）が増えれば増えるほど，その組み合わせ分のディレクトリを用意しファイルを各々のディレクトリにコピーしてサブセットを作っておかなければならない。TXTANA なども同一のツリー下ならばまとめて扱えるが，兄弟の関係にあるディレクトリの場合には別サブコーパスとして登録してあとで組み合わせなければならない。学習者コーパスの場合には比較する変数が多いので，登録するコーパスの数もなん十個にもなってしまって不便なのである。

図5：ディレクトリ管理の欠点

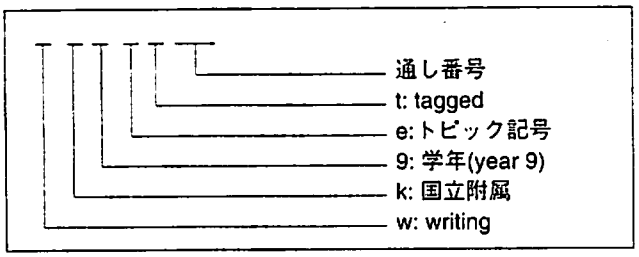


そこで第2段階としては，ファイル名を識別子に使うという方法がある。ディレクトリに依存した管理方法では，そのディレクトリ下のファイル名は極端な話，どうでもよいことになる。異なるディレクトリであれば，001.txt という同一名のファイルがあっても支障はない。しかし，大量のデータを扱い始めると，徐々にこのようなディレクトリ管理だけでは不十分なことがわかってくる。このような場合に備えて，ディレクトリ管理をするのと同時にファイル名も工夫して付けておくといいだろう。ファイル名を付ける際には，

(a)ファイルの拡張子は .txt または .xml/.cha などフォーマットに合わせて付ける
(b)ファイル名は8文字以内にする（MS-DOSの制限のため）
(c)ファイル名を一意に特定できるように付ける

といったことがポイントだ。たとえば，私の管理している learner data は wk9et001.xml のようなファイル名を付けている。図6に示すようにそれぞれが分類の記号になっているので，ファイル名を指定する際にワイルドカードを使っ

図6：ファイル名の付け方の例



てフィルタが可能だ。

第3段階のファイル管理として，前回紹介したような1つ1つのファイルにヘッダ情報を盛りこんだ形式を利用することをお勧めする。こうすることで，コーパスそのものに付加価値として学習者情報を盛り込むことができ，ディレクトリ管理だけではどうしても限界だった研究デザイン上必要な変数をテキスト内にデータとして豊富に盛り込むことが可能になる。それを XML などの，情報の構造を明示できるようなフォーマットで管理することによって，コーパスデータがデータベースのように利用できるようになるのである。

私は用途に応じてこの3種類のファイル管理方法を混在させて使っている。UNIX で作業をする際には，ファイル名での操作が圧倒的に多いので，ファイル名でフィルタしてパイプ処理（複数のコマンドを連続して処理すること）などをする。国産コンコーダンサーの TXTANA は細かい絞り込み処理などをする際に威力を発揮するので，ディレクトリ主導の管理用の基本セットも用意しておく。XML 形式で処理する方法はまだコーパス言語学では研究途中なのだが，研究用としてデータを構造化するために1つの XML ファイルと DTD で数千人のデータを管理する方法も考えたりする。ぜひ，読者諸兄も参考にして自分に合ったファイル管理の方法を身に付けていただければ，と思う。

さて，XML およびファイル管理に関してはこのくらいにして，いよいよこのようなフォーマットで作成した学習者コーパスのファイルをもとに，コーパスの検索の実際を WordSmith というコンコーダンサーを用いて紹介しよう。前回紹介したさまざまなコーパスのフォーマットではどれもヘッダ情報を明記しているものが多かったが，WordSmith でそのような情報を生かしてどのような操作が可能か注目しながら読んでみていただきたい。

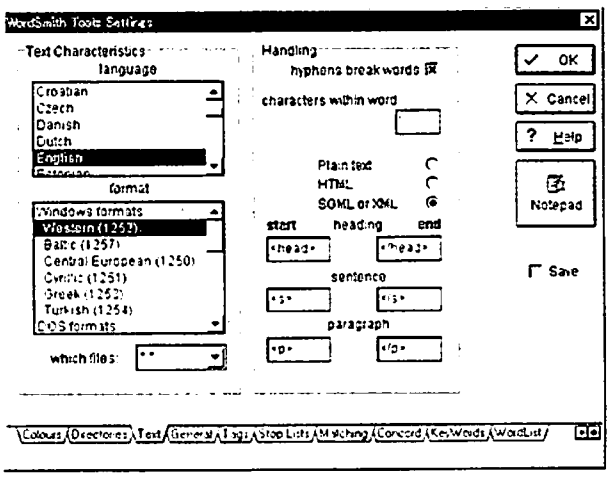
WordSmithを用いたファイル操作

WordSmith は Liverpool 大学の Mike Scott が開発し，OUP が販売している市販のコンコーダンサーである（URL:http://www1.oup.co.uk/elt/catalogue/Multimedia/WordSmithTools3.0/download.html）。現在，ユーザー・フレンドリーなコンコーダンサーとしてはもっとも高機能なものの1つだ。上記 URL からプログラムをダウンロードしてきて試用することができる。気に入ったら OUP に送金し，upgrade すればいい。

タグ付きファイルの認識

WordSmith はバージョンアップするごとにタグ付きコーパス・データへの対応を積極的に進めており，現行の version 3 でもすでに plain text, HTML, SGML/XML の3種類のフォーマットに対応している（多少動作が不安定なところがある）。図7が WordSmith の設定画面であるが，ファイルの Handling のところを SGML or XML に設定すると，heading, sentence, paragraph といった要素に必要なタグが認識されるようになっている。これによって，通常のテキスト・データでは認識させにくい sentenceの境目などの情報も適確に処理することができるようになっている。

図7



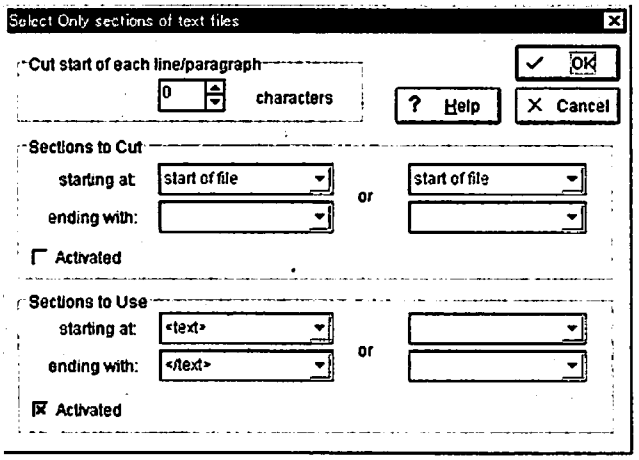
検索テキスト部分の指定

また，XML, CHAT, COCOA のようにヘッダ情報が付

いている場合には，実際の処理ではそのヘッダ部分は除外して検索を行わなければならない。WordSmith ではそのような指定も細かくできるようになっている。設定画面の[Tags]というオプションで実際に検索するテキスト部分を指定してやれば，単語検索やリストの作成の際にヘッダ部分は無視して検索をしてくれる。図8では<text>タグの部分だけ検索するように指定している。

さらに WordSmith では付属の viewer で XML のようなタグ付きファイルを閲覧することができる。Version 3 では viewer の機能の一部がまだ未完成なので時々ハングしたりするが，Version 4 では XML 対応をさらに強化する予定らしいので，今よりもずっと動作が安定してタグの on／off をしながらファイルの閲覧などができるようになるはずだ。

図8



コーパス・データ検索の基本的な考え方

さて，WordSmith を使ったいろいろな検索事例を見る前に，大まかにコーパス・データ検索の基本的な考え方をまとめておこう。

研究デザインとコーパス・データの関係

まず第一に，コーパスによる研究を行う際には，第2回で解説したように，ある程度事前に検討された研究デザインとその処理の結果として出てくるコーパス・データの対応関係が必要だ。ただ自分のクラスのある授業でなんとなく行った英作文の課題をコンピューターに入れてその語彙リストを作った，という程度ではたくさんの語彙統計は出てきてもおもしろい研究にはならない。コーパスを使ってやろうとしていることの目的がはっきりしていないからである。

たとえば，一般のコーパス言語学ではよくアメリカ英語とイギリス英語のコーパスを比較したり，年代の異なる英語のコーパスを比較したりする。これも，コーパス・データの収集自体は自分ではしないけれども，「英語の変種」であるとか「歴史的変化」などといった研究上興味のある「変数」を研究者が設定して，その変数で調整した結果与えられる異なる特徴をもったコーパスを選んである言語の側面について比較しているわけである。

学習者コーパスにおける研究でも変数としてはいろいろなものが考えられる。学習者内要因(learner-internal factors)を考えると，年齢，性別，母語，学習スタイル，認知スタイル，動機づけ，などが考えられるし，学習者外要因(learner-external factors)では個々の指導法の影響や外国語に触れる量，その外国語の社会的な位置づけなど，さまざまな要因が考えられる。そういった意味では，研究デザインとコーパス・データとの関係は一般的な研究デザインとそれによって得られるデータの関係に等しい。しかし，得られる学習者データとして，他の方法（例:単語テストや読解テストなど）をさせずに発話データや作文データをコーパスにして分析するというからには，コーパス・データによる分析に適した研究対象を選ぶ必要がある。

図9は，研究者が見たいと思う変数を独立変数(independent variables）として設定して，その変数の影響を受ける部分を従属変数（dependent variable，たとえば英語力の伸長など）としてコーパスデータで捉えようというアプローチである。この際，異なる変数（または同一の変数の異なるレベル）によって分けられた各グループからデータを取り，各データをコーパス分析した結果を統計的に比較するわけである。

図9に対して，もう1つ別のアプローチが可能だ。それは図10にあるように，同一コーパス内で複数の比較可能な特徴を抽出してその間の関係を探るという方法である。たとえば，British National Corpus のように大規模なコーパスになれば，成人母語話者の英語データとして

図9　研究デザインとコーパスデータの関係（1）

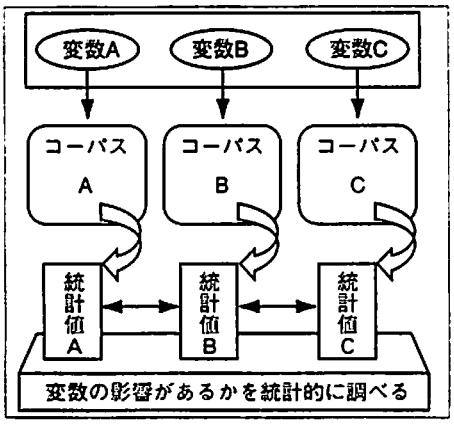
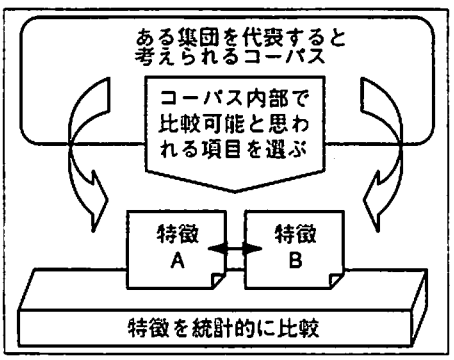


図10　研究デザインとコーパスデータの関係（2）



はかなり一般性を持っていると仮定して，その written corpus の中で関係節の制限用法と非制限用法の頻度を調査し，その文脈における関係節の働きなどを詳細に分析したりすることができる。

同様の例は，学習者コーパスであれば，大学生ならばあるまとまった量の大学生の日本人英語学習者コーパスをもとに，前置詞の正用法と誤用とをタグ付けして，前置詞のタイプによって誤用率は変化するか，どのような誤用が多いかといった分析をしたりできるだろう。

もちろん，この2つのアプローチは決して相容れないものではなく，実際はかなりの研究で2つのアプローチを組み合わせたデータ分析の方法が用いられている。

コーパスではわからないことをわきまえる

連載第1回目から強調していることだが，コーパスは「道具」であり「データ」の1種である。コーパスは確かに従来の研究ではなかなかわからなかった言語使用のさまざまな側面を浮き彫りにしてくれる強力な武器であるが，と同時に「万能薬」ではない。むしろ，コーパスではわからないことに関してきちんと理解しておく必要がある。次回から具体的な学習者データの検索を試みる際に，「コーパスではわからないこと」を簡単にまとめて今回の解説を締めくくりたい。

①「文法性の判断」は苦手

母語話者の直観ではある文が「可能かどうか」を比較的容易に判定することができるが，コーパスにはすべての文がコーパスに出てくるわけではないので，文法性の判断は苦手である。とくに生成文法の研究などで盛んに用いられる「非文」の検証はコーパスでは不可能なことが多い。

②Avoidance の判断がむずかしい

言語使用で，母語話者に対して学習者の overuse, underuse がわかるということが，学習者コーパスの強みであるが，一方で言語使用のデータには avoidance による「使わない」ものに対してはデータが取れない，という弱点がある。

③言語の受容的知識(receptive knowledge)に関してはわからない

言語使用には comprehension のほうも当然重要な力として考慮すべきだが，コーパスでは話者の受容的知識に関してはまったくわからない。

以上のような弱点をもっと掘り下げて考えてみたいが，紙数が尽きたのでまたの機会にしたい。いよいよ次回は，WordSmith を使って具体的にどのようなデータの分析が可能かをじっくり見てみることにしよう。

参考文献
「標準XML完全解説」(XML/SGMLサロン著) 技術評論社，1998.
*XML の親切な概論書。

有益な web ページ
■"XML and the Second Generation Web" by John Bosak and Tim Bray (http://www.sciam.com/1999/0599issue/0599bosak.html)
*XML の秘める力を開発者が語っている。
■WordSmith Tutorial Webs (http://www.lancs.ac.uk/postgrad/tono/wsmith/index.html)
*筆者のホームページの中でWordSmithの具体的な使い方をlearner data を例にして解説している。

問い合わせ：y.tono@lancaster.ac.uk