『より良い外国語教育研究のための方法』(pp. 74–99)
外国語教育メディア学会 (LET) 関西支部 メソドロジー研究部会 2010 年度報告論集

学習者コーパス入門
―NICE を利用して―

阪上 辰也
名古屋大学

概要

本稿の目的は，学習者コーパス「Nagoya Interlanguage Corpus of English」（略称：NICE）を用いて，学習者コーパスの基本的な利用法を説明することである。

「学習者コーパス」は，外国語を学習する人が，学んでいる外国語で書いたり話したりしたものをテキストとして集めたものであり，世界的には，英語を学ぶ人を対象にした学習者コーパスが多く作られている。しかしながら，こうしたコーパスの利用法に関する情報が乏しいという現状がある。そこで，本稿では，NICE を具体例として，汎用的に応用可能な言語処理技術を中心に，コーパスの構築手順やデータの処理方法・分析方法について解説する。

キーワード: 学習者コーパス, NICE，正規表現

1. 学習者コーパスとは何か

コーパスとは，「大量のテキストを集め，“コンピュータで処理できる状態”になったデータベース」のことである。「コンピュータで処理できる」という点が重要であり，これまでは本などの資料を読んで手作業で調べていた用例などを，コンピュータを利用することで，瞬時に調べられるようになった。また，調べた表現がいくつ観察されるかという頻度を計算でき，頻繁に使用する表現かそうでないかを容易に知ることができる。

コーパスには，様々な種類がある。英語について研究する際に最も利用されるのは，英語を母語とする人が書いたものや話したことを集めたコーパスである。例えば，British National Corpus（http://www.natcorp.ox.ac.uk/）は規模が大きく，約 1 億語の英語データが収録されている。この他にも，新聞記事を集めたコーパスや，友人が話した内容を録音し，それを書き起こしたコーパス，E-mail を集めたコーパスも存在する。中でも，2001 年に不正会計事件を起こしたアメリカの企業「エンロン」社の E メールを集めた「Enron Corpus（http://www.cs.cmu.edu/~enron/）」は有名で，無償でデータを入手できる。このように，コーパスには多くの種類が存在するが，本稿では，「学習者コーパス」（Learner Corpus）というコーパスを取り上げる。

「学習者コーパス」は，外国語を学習する人が，その学んでいる外国語で書いたり話したりしたものをテキストとして集めたものであり，世界的には，英語を学ぶ人を対象にした学習者コーパスが多く構築されている。

本稿では，科研費プロジェクト[1]の成果の一部として 2008 年 10 月に公開された学習者コーパス「NICE」という学習者コーパスを取り上げる。NICE は，「Nagoya Interlanguage Corpus of English」の頭文字を取ったもので，名古屋圏を中心にして作られたことに由来して名付けられた。この NICE の中身がどのようになっているのか，どのような手順で構築されるのか，どのように検索したらよいかなどを中心に，学習者コーパスの基本的な利用法について説明する。

1.1 学習者コーパスの定義

Nesselhauf（2004）は，学習者コーパスを「systematic computerized collections of texts produced by learners」と定義している。「学習者により産出されたテキストを体系的で電子的なものとして集めたデータ」という意味になるが，重要な点は，「体系的」であることと「電子的」であることの 2 点である。要点は，以下のようにまとめることができる。

・体系的：条件（作文，インタビューなど）を設けてタスクとして産出させる
・電子的：コンピュータで処理できる状態になっていること

まず，「体系的」というのは，様々な条件を統一した上で産出を行い，データを収集することである。様々な条件を統一するというのは，例えば，作文（書き言葉）なのか，インタビューなどで発した言葉（話し言葉）なのかというデータの種類を統一する，あるいは，どのような習熟度の学習者を対象にして集めるかというデータ提供者を統一するということである。データを分析し，その結果に対する考察を行う上で，産出条件の統一は必要である。

次に，「電子的」というのは，コンピュータで処理ができるように，データの記録形式などを決めておくということである。コンピュータを使って人がデータを読むことができればそれでよい，という意味ではない。データの中には，学習者が産出したデータだけでなく，学習者自身に関する情報（性別，年齢，英語の学習歴などの属性）も含めて記録することが多く，これらの情報をどのような「決まり」で記録するかを予め決めておく必要がある。「決まり」がなければ，データを適切に処理することができず，誤った結果（単語の頻度を数え間違えるなど）を導いてしまうことにつながることがあるため，必ず「決まり」，つまり，データの「フォーマット」を決定する必要がある。フォーマットについては，2.2 節で詳述する。

1.2 学習者コーパスは何に使われるか

学習者コーパスが，どのような分野で利用されているかを簡単に紹介する。学習者コーパスは，主に 2 つの分野で利用されており，1 つは「第二言語習得研究」，もう 1 つは，教材開発である。

1.2.1 第二言語習得研究

第二言語習得研究は，人が母語とは異なる言語をどのように学んでいるのかという，その仕組みを明らかにしようとする分野である。学習者コーパスを利用した研究では，中間言語（ある外国語を学んでいる段階で，学習者に特有な特徴を持った言語のこと）の分析が頻繁に行われる。この中間言語の分析のことを，「Contrastive Interlanguage Analysis」（Granger, 2002）と呼ぶ。この中間言語分析には，主に 2 つの比較パターンが存在する。

a) 母語話者 vs. 学習　　：中間言語の特徴は何か
b) 学習者 vs. 学習者　　：中間言語同士で共通する・異なる特徴は何か

まず，母語話者と学習者という比較により，中間言語の特徴が何であるかを明らかにすることができると考えられている。例えば，ある単語の信用頻度の差を比較し，母語話者よりも学習者の方が多く産出（過剰使用）していることがわかれば，学習者は，その単語の使用については，不自然な使い方をしている可能性があると考えることができる。

次に，学習者と学習者という比較では，中間言語同士を比較し，中間言語間で何が共通するか，母語の影響はあるのかを調査する際に用いられる比較パターンである。具体的な調査対象としては，定型表現（collocation），接続語句，誤用の傾向，品詞の連鎖などを分析し，例えば，日本人英語学習者に特徴的な表現を明らかにすることができる。また，それぞれの母語が異なることから，中間言語間で異なる特徴が出た場合に，その要因のひとつとして母語の影響を予測することができる。言い換えれば，母語の特徴が，中間言語にどれほど影響を及ぼすのかという研究も可能になる。

1.2.2 教材開発

学習者コーパスは，教材開発の分野でも利用され，間接的，および，直接的に利用されることがある。

・間接的：辞書への記載（間違えやすい語法の確認）
・直接的：学習者が見て，誤用を確認（母語話者コーパスとの比較）

「間接的」な利用とは，学習者コーパスの分析結果から得られた知見を教材作成に活かすという意味である。例えば，discuss という他動詞の後に，about のような前置詞を使用する誤用例がコーパスから見つかった場合，その結果を踏まえて，辞書の記述内容を変更し，間違えやすいので注意が必要という形になるというケースが考えられる。

「直接的」な利用とは，学習者自身が学習者コーパスと母語話者コーパスを比較分析し，何が誤用かを独力で発見させる活動のことを意味する。直接的な利用では，お手本としての母語話者コーパスが必要になり，その両者を分析することが必要になるため，分析スキル，および，文法等の知識の習得が不可欠となる。そのため，実際の利用例は多くないものと思われる。

1.3 世界にはどのような学習者コーパスがあるか

本節では，本稿で中心的に取り扱う「NICE」も含め，主に 4 つの学習者コーパスの概要を紹介する。

1) ICLE（International Corpus of Learner English）
2) The NICT JLE Corpus
3) JEFLL Corpus（Japanese EFL Learner Corpus）
4) NICE（Nagoya Interlanguage Corpus of English）

まず，「ICLE」は，異なった言語背景を持つ英語学習者の書き言葉を集めたコーパスである。データを構成する学習者の母語は，14 言語に及び，中間言語同士の比較が可能となる。規模は 200 万語の作文データであり，学習者コーパスとしては，世界一の規模であると言える。また，比較のために母語話者コーパスも用意されており（「統制コーパス」と呼ばれる），中間言語に特徴的な言語表現を観察することも可能である。

次の「The NICT JLE Corpus」は，日本人英語学習者の発話データで，規模は約 100 万語である。発話の内容から，レベル分けがなされており，習熟度別の比較が可能となる。つまり，習熟度が上がるにつれて，どのような表現が話せるようになるのか，また，習熟度により，誤用の量や質にどのような変化が見られるのか，といった分析をすることができるようになる。

続いて，「JEFLL Corpus」は，日本の中学・高校生英語学習者による自由英作文コーパスであり，1 万人分のデータ（67 万語）が収録されている。オンラインでの検索（http://scn02.corpora.jp/~jefll03/jefll_top.html）が可能になっており，各学年でのデータの絞り込みや，品詞情報に基づいたデータ抽出ができる。

最後に，本稿で取り上げる「NICE」は，日本人大学生・大学院生の作文データを収録しており，規模は約 7 万語（学習者のデータのみ）となる。ICLE に比べると，収録語

数が少ないが，NICE は，書き手の属性が ICLE よりも豊富に記録されており，統制コーパスだけでなく，母語話者による添削文も付与されており，これらの点で，NICE は ICLE とは異なった研究に有用な学習者コーパスであると言えるだろう。

2. 学習者コーパス「NICE」の特徴

本節では，NICE（Nagoya Interlanguage Corpus of English）の特徴について説明する。NICE は，2 つのコーパス（サブ・コーパス）から構成されている。1 つは，日本人英語学習者（大学生と大学院生）により書かれた英作文が含まれたコーパスであり，207 名分のデータが含まれている。非母語話者（Non-Native Speaker）が書いた作文データであることから，「NICE-NNS」と呼ぶ。もう 1 つは，英語母語話者が書いた英作文が含まれたコーパスで，200 名分のデータが含まれている。こちらは，母語話者（Native Speaker）が書いた作文データであることから，「NICE-NS」と呼ぶ。

これら 2 つのコーパスから成る NICE には，以下の 3 つの特徴がある。次節以降で，3 つの特徴について説明する。

① 詳細な属性の記録
② データの読みやすさ
③ 母語話者による添削文の付与（NICE-NNS のみ）

2.1 詳細な属性の記録

まず，特徴①の詳細な属性の記録について説明する。実際には，「学習者」と一括りにして分析することは難しく，様々な学習者を考慮した分析が必要となる。そのため，どのような学習者がどのような作文を書いているのかを分析するには，作文をした者が，どのような人かを記録しておく必要がある。NICE-NNS では，a) 年齢，b) 性別，c) 英語の学習歴，d) 英語以外の外国語の学習歴，e) TOEIC や英検などの英語の資格，f) 日常での英語の使用状況，g) 英語で文章を書くことに対する自信度といった多様な情報が記録されている。

例えば，NICE-NNS の場合には，207 ファイルのうち，121 ファイルに TOEIC スコアが記録されている。その TOEIC スコアを基準としてデータを分類すれば，TOEIC スコアで高得点を取った学習者とそうでない学習者を比較することができるようになる。他にも，日常での英語の使用状況という情報を活用することにより，「英語の使用状況が多ければ多いほど，たくさんの英文が書けるようになるか」という研究課題を設定してコーパスを分析することもできる。

また，NICE-NS でも同様に，年齢や性別に加えて，どの国の英語か（アメリカ・イギリス・カナダ・オーストラリア），両親の母語，最終学歴，外国語の学習歴，書くこと

に対する自己評価といった情報が記録されている。「英語母語話者」と一括りにするのではなく，どのような背景を持った英語母語話者が書いているのかを知ることができる。

このようにして，NICE には，どのような人が作文を書いたかが記録されており，様々な側面から作文を分析することができるようになる。次節では，そうした学習者の属性や作文のデータがどのように記録されているかという「決まり」，つまり，「フォーマット」について説明する。

2.2 データの読みやすさ

特徴②の「データの読みやすさ」について説明する。一般に，データは，一定の「決まり」に従って記録される必要がある。つまり，どこにどのような情報を配置しているかを分かるようにしておくということである。決めたデータの記録形式のことを，「フォーマット」と呼ぶ。NICE の構築に際しても，データのフォーマットを決めている。NICE のフォーマットを決めるにあたり，2 つの点が配慮されている。1 つは，コンピュータによる「データ処理の効率を上げる」こと，もう 1 つは，人がデータを読む際の「可読性」を高めることである。まず，前者の「データ処理の効率」について説明する。

学習者コーパスの場合，各学習者が，どれほど作文を書いたのかを分析することがある。例えば，ある学習者が，制限時間内（NICE の場合，作文時間は 1 時間に制限されている）にいくつの文を書くことができたか，という数値を求める場合を考えてみる。この時，文の数という数値を求めるために，予め，「1 行に 1 文」の形でデータを記録しておくと，分析の効率が上がる。実際の NICE のデータを図 1 に示す。

```
@Begin
@Participants:      JPN001
@Age:  20
@Sex:  F
@YearInSchool:      U2
@Major:      law
@StudyHistory:      8
@OtherLanguage:     Chinese=1;none=
@Qualification:     STEP=2(2001);none=;none=
@Abroad:     none=;none=
@Reading:    3
@Writing:    2
@Listening:  1
@Speaking:   1
@JapaneseEssay:     2
@EnglishEssay:      2
@Difficulty: 5
@Topic:      death penalty
@Comments:
@Coder:      2005-12-21 DataInputBy SAKAUE Tatsuya;
@Version:    1.1
*JPN001:     Does death penalty really need?
%NTV:  Is the death penalty really necessary?
%COM:
%par:
*JPN001:     Does death penalty really need?
%NTV:  Is the death penalty really necessary?
%COM:
*JPN001:     I often hear that death penalty is reasonable for people whose
family member or friend is killed by someone.
%NTV:  People often say that the death penalty is reasonable in the eyes of
people whose family members or friends have been killed by someone.
%COM:
```

図 1.　NICE のデータの一部

学習者のデータは，行の最初の部分に「*JPNxxx:」と書かれた行に記録されている。したがって，この作文の書き手が書いた文の数を求めるには，「*JPN」で始まる行がいくつあるのかを数えればよいということになる。Microsoft Excel などの表計算ソフトを使う場合は，オートフィルタ機能を使い，「*JPN」が含まれるデータのみを表示すれば数を求めることができる。また，Mac OS X や Linux ユーザーであれば，grep コマンドと wc コマンドを組み合わせることで，行数を求めることができる。このように，データ処理の効率を上げるため，NICE では，1 行に 1 文が並ぶ形式で，学習者の作文データが記録されている。

もし，このデータが，「1 文に 1 行」という形に整形されていなかったとすると，文の切れ目（「.」「!」「?」などの記号類）を目視で探し出し，1 つの文に分解する作業から始めなければならない。NICE では，そうした手間を予め省き，効率よくデータ処理ができるような状態でデータが保存されている。

続いて，2 つの「読みやすさ」のうち，「人から見た読みやすさ（可読性)」について説明する。

最近のコーパスには，人が発したことばのデータに様々な情報を加えられている（annotation と呼ばれる)。特に，その単語の「品詞」(Part of Speech (POS)）が何であるかという情報が記録されることが多く，その際に「タグ」が用いられ，近年の傾向として，

XML（eXtended Markup Language）形式が多く採用されている。具体例として，タグを利用して品詞情報を付与したデータの例を図 2 に示す。

```
<w c5="DT0" hw="this" pos="ADJ">This </w><w c5="VBZ" hw="be" pos="VERB">is
</w><w c5="AT0" hw="an" pos="ART">an </w><w c5="NN1" hw="apple"
pos="SUBST">apple </w><c c5="PUN">.
```

図 2. タグ付けされたデータの一部

図 2 は，British National Corpus（BNC）でも採用されているフォーマット（XML 形式）で記録されたデータである（実際の BNC のデータではなく，仮の英文データにタグを付けている)。元の文は，「This is an apple.」という簡単なものであるが，単語の両隣にある＜・＞という不等号で囲まれた部分に品詞情報が付与されており，それらの情報が邪魔をして，何が元々書かれていたのを読み取ることは困難だと思われる。図 1 と見比べると，読みやすさが確実に低下していることが分かる。

NICE には，行頭に「*JPNxxx」という目印があり，加えて，1 行に 1 文というフォーマットが決まっているため，学習者が何を書いていたのかを，容易に読み取ることができる。学習者コーパスの分析では，単語や文などがいくつあるかという頻度を求めるだけでなく，どのような内容の文を書いていたかという意味解釈を行い，質的分析を加えることも必要になる。意味解釈・質的分析を行うのは人であり，その人が読みやすい状態で記録しておくことも必要になる。

なお，コーパスの構築においては，NICE のように，タグなどによる情報の付与がなされていないコーパス（raw corpus）と，情報の付与をしたコーパス（annotated/tagged corpus）の 2 種類を用意しておき，研究調査の目的に応じて使い分けることが望ましい。

2.3 母語話者による添削文の付与

特徴③「母語話者による添削文の付与」について説明する。学習者が，英語で書いたり話したりするものの中に，何らかの「エラー（誤用)」が含まることがある。簡単な例を挙げると，「He lives in Nagoya.」という文を書くべきところを，ある学習者が「He live in Nagoya.」と書いた場合，この文には，文法に関するエラーが含まれている。つまり，三人称単数形である主語の He に対応して，動詞の現在形は，語尾に s を付けて「lives」と書くべきだが，その動詞を変化させなかった，というエラーになる。こうしたエラーに関する情報と，「正しくはどのように書くべきだったのか」という修正内容については，これまでの学習者コーパスには十分に付与されていない。

そこで，NICE の構築にあたっては，英語母語話者による添削文を付与することにした。添削文を付与することにより，どのようなエラーが含まれていたか，実際はどう書くべきだったのかを分析できるようになる。図 3 を参照されたい。

```
@Begin
@Participants:      JPN001
@Age:  20
@Sex:  F
@YearInSchool:      U2
@Major:       law
@StudyHistory:      8
@OtherLanguage:     Chinese=1;none=
@Qualification:     STEP=2(2001);none=;none=
@Abroad:      none=;none=
@Reading:     3
@Writing:     2
@Listening:   1
@Speaking:    1
@JapaneseEssay:     2
@EnglishEssay:      2
@Difficulty:  5
@Topic:       death penalty
@Comments:
@Coder:       2005-12-21 DataInputBy SAKAUE Tatsuya;
@Version:     1.1
*JPN001:      Does death penalty really need?
%NTV:  Is the death penalty really necessary?
%COM:
%par:
*JPN001:      Does death penalty really need?
%NTV:  Is the death penalty really necessary?
%COM:
*JPN001:      I often hear that death penalty is reasonable for people whose
family member or friend is killed by someone.
%NTV:  People often say that the death penalty is reasonable in the eyes of
people whose family members or friends have been killed by someone.
%COM:
```

図 3.　NICE のデータの一部（図 1 を再掲）

NICE では，母語話者による添削文が，「%NTV」（NTV は，native を略記したもの）で始まる行に書き込まれている。例えば，1 つ目の文では，「Does death penalty really need?」という学習者が書いた文を，母語話者は「Is the death penalty really necessary?」と書き換えている。この学習者の文は，他動詞の need を使用した結果，「死刑が，本当に（何かを）必要としているか」という文になってしまい，意味が通じない。そこで，学習者の意図は，「死刑という制度は本当に必要なものであるのか」と述べることにあると考えられ，母語話者は，be 動詞と形容詞の necessary を使って書き換えているのである。

なお，添削にあたっては，「学習者が使っている単語や表現をできるだけ活かす形で書き換える」という方針を母語話者に伝えている。もし，方針に「従っていない」添削文，つまり，学習者の書いた単語や表現をほとんど活かしていない添削文になっていれば，学習者が書いた単語や表現に深刻なエラーがあるのではないかと推測することができる。

最後に，添削文の利用について，気をつけるべき点が 2 つ述べる。1 つは，母語話者による添削文は 1 つのみという点である。添削文は，唯一の絶対的な正用ではなく，別の書き換え方の可能性があるということを心に留めておく必要がある。もう 1 つは，分析が必要であるという点である。「%NTV」の行には，添削文のみが記録されている状態であり，それがどのようなエラーで，どの部分が何に書き換えられているかについては，分析者がエラーの分類・判断を行い，集計しなければならない。

3. NICE を実際に検索する

本章では，どのような操作によってデータを加工・検索するのか，その具体的な手順を説明する。

3.1 データのダウンロード

まず，NICE のデータをダウンロードする方法を説明する。手順は以下の通りである。

① Web ブラウザ（Firefox, Chrome, Safari, Opera, Internet Explorer など）を起動して，http://sugiura5.gsid.nagoya-u.ac.jp/~sakaue/nice/ にアクセスする。
② 左のメニューから「NICE のダウンロード」という項目をクリックする。
③ 注意事項などを読み，「NICE (ver.*バージョン番号*)のダウンロードページへ」というボタンをクリックする。
④ 「NICE (ver.*バージョン番号*)のダウンロード」というボタンをクリックする。
⑤ 保存先をデスクトップに指定し（参照：図 4），「保存」ボタンを押す。

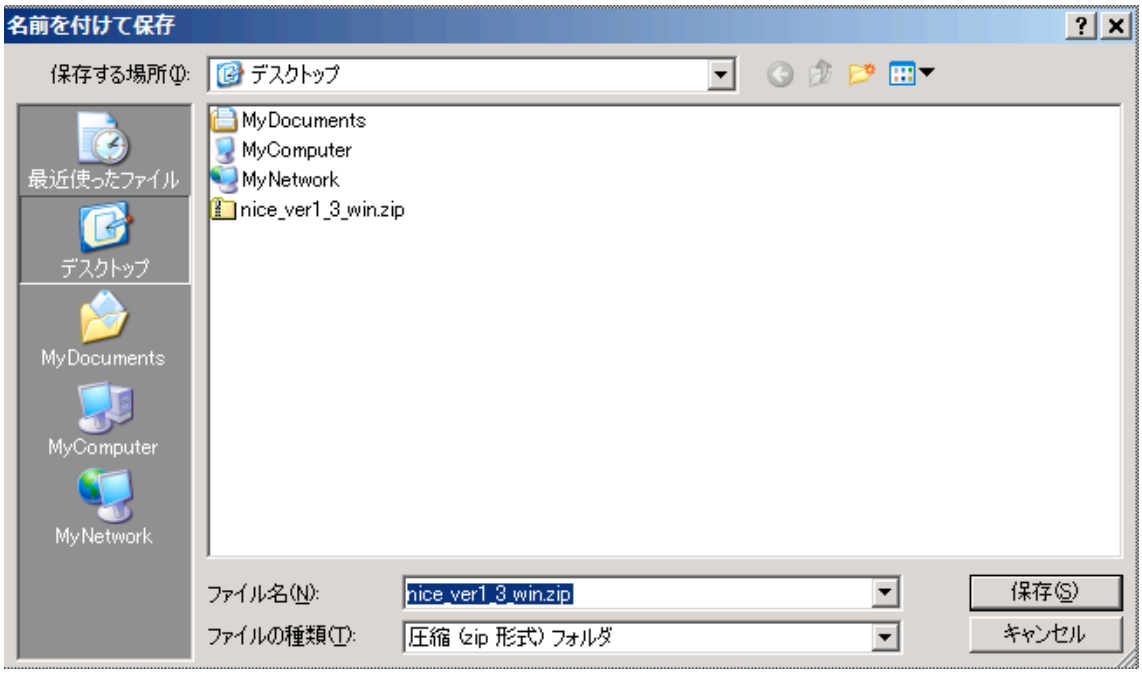

図 4.　NICE のデータをデスクトップに保存するように指定した画面

ダウロードが成功していれば，デスクトップに，「nice_ver_x_x_x.zip」（x は任意の数字）という名前のファイルのアイコンが表示される。

次に，ダウンロードしたファイルの展開を行う。ダウンロードしたファイルは，圧縮されたファイルであるため，そのままでは読み取ることができないため，圧縮されたファ

イルを「解凍」（読み取り可能な状態に戻すこと）する必要がある。圧縮ファイルの解凍の手順は次の通りである。

① 圧縮ファイルをダブルクリックする。
② 圧縮ファイルの中身が表示され，フォルダが現れるので，そのフォルダをデスクトップにドラッグ＆ドロップする（参照：図 5)。

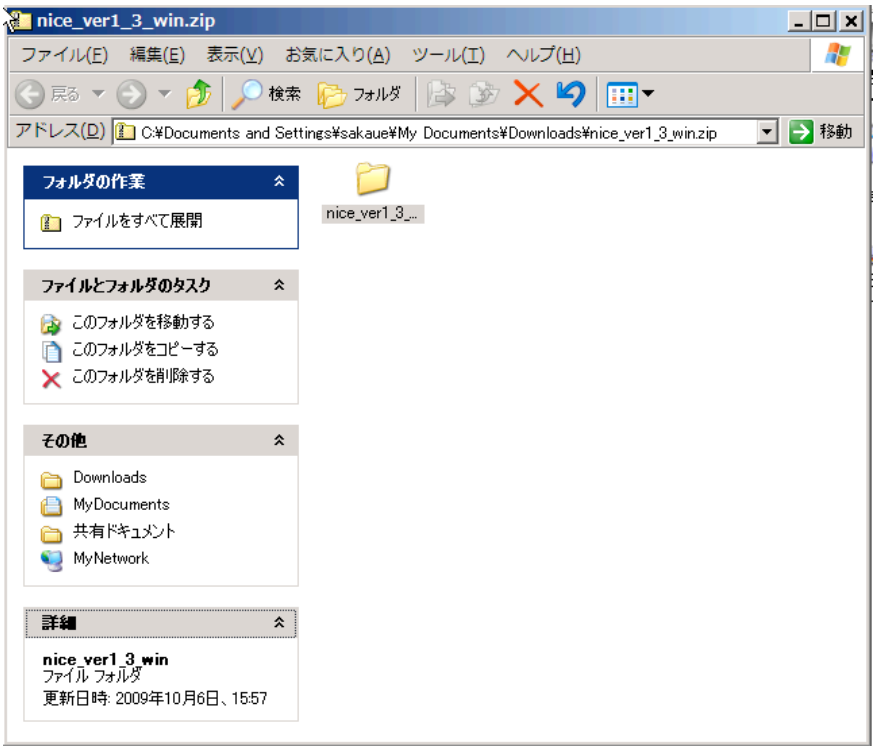

図 5.　圧縮ファイルをダブルクリックした直後の画面

なお，フォルダをドラッグ＆ドロップするという方法の他に，図 5 で確認できるように，左のメニューから「このフォルダをコピーする」という項目を選び，コピー先をデスクトップに指定して保存することも可能である。

3.2　検索（基礎編）

3.2.1　“I think”を検索する

今回は，基本的な検索の方法について説明するが，ソフトウェアエアとして，「サクラエディタ」（http://sakura-editor.sourceforge.net/）を利用する。このサクラエディタには，「Grep」という検索機能があり，検索した表現の一覧を表示することができる。なお，一部の操作については，操作の模様を記録した動画（スクリーンキャスト）を用意しているので，適宜参照されたい（http://www.screencast.com/t/4grN1KXv)。

検索を行うための操作手順として，

① メニューから「検索」を選び，その中の「Grep」をクリックする。
② 条件（＝検索したい表現），検索するファイルと検索するフォルダを指定する。
③ 検索する表現を入力し，「検索」のボタン押す。

という 3 段階の操作がある。

操作手順②にある，検索するファイルの指定では，NICE のデータはすべて，XXX.txt の形式で保存されているため，ファイルの欄には，「*.txt」と入力する（「*」は，どのような文字でもよいことを示す「正規表現」と呼ばれるものだが，正規表現については後述する）。

また，検索するフォルダについては，NICE の入っているフォルダを選択する。なお，動画では，NNS のデータが入っているフォルダのみを検索対象にしている（参照：図 6）。

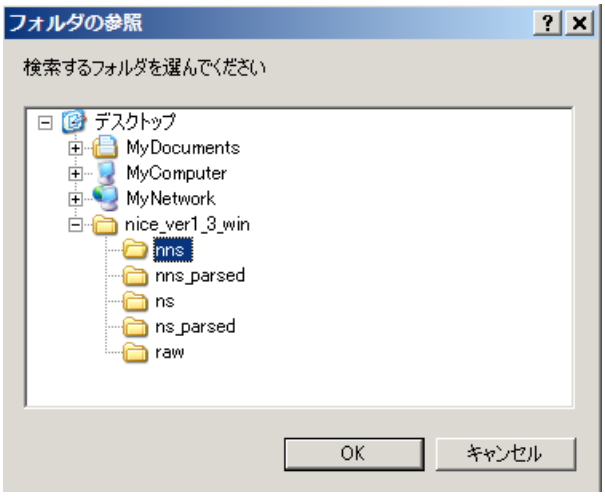

図 6.　フォルダの指定（nice_ver_1_3_win の中の nns 内を対象とする場合）

動画では，「I think」という表現を検索している（検索の条件指定は図 7 を参照）。「検索」ボタンを押してから，「I think」を含んだデータが，検索結果として表示される（参照：図 8）。「検索したいものを入力して，検索ボタンを押す」ことが，サクラエディタの基本的な検索方法となる。注意点は，どのフォルダ・ファイルを対象に検索をするかの指定であり，指定を誤ると，正しく検索できず，結果が表示されないため，検索を実行する前に，しっかりと確認する必要がある。

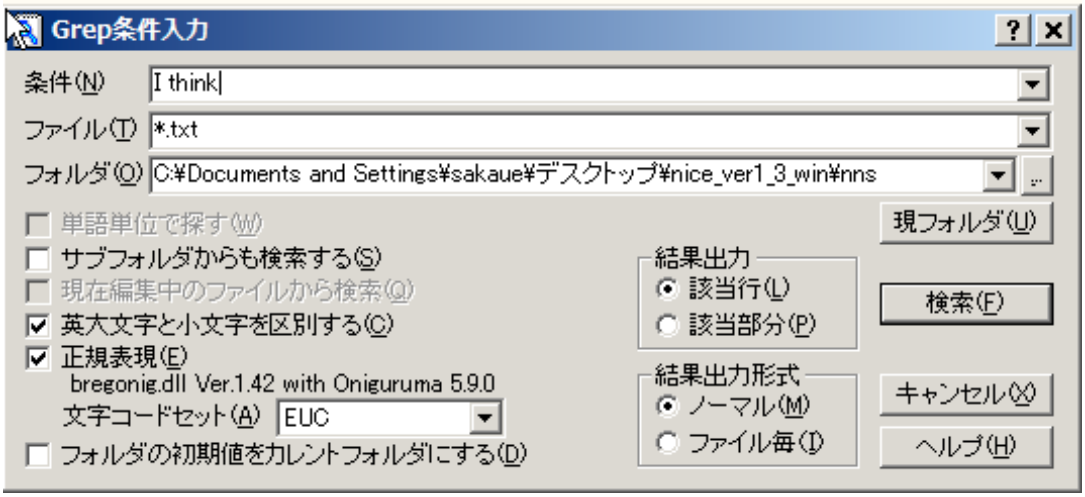

図 7.　検索条件の指定画面

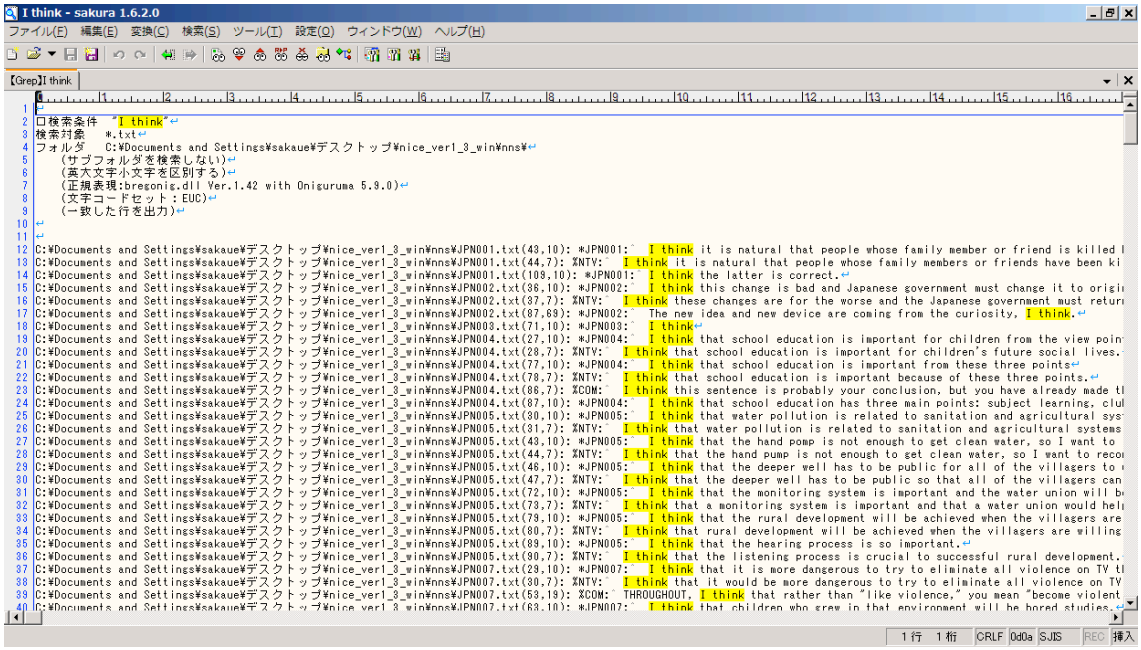

図 8.　検索結果の表示画面

検索結果の画面を一番下までスクロールすると，559 件がマッチしたことを確認することができる。

3.2.2　“computer”と“computers”を同時に検索する

コーパスを検索する際に「正規表現」を知っていると，検索を効率よく行うことができる。正規表現とは，「文字の並び（パターン）を，より少ない文字列で表す表記法」のことで，言い換えれば，「少ない労力で，できるだけ多くの検索をまとめて実行する」ための表記法である。例えば，コーパスから「computer」という単語を検索する場合，

「computer」には複数形の「computers」もあることから，両方を検索対象に含める必要がある。2 つの単語をサクラエディタで検索する場合，「computer」と「computers」の 2 つを個別に入力して，2 回の検索を行い，検索数を合計するのも一手であるが，正規表現を使った場合，この 2 つを記号混じりの「computer[s]?」という 12 文字で表すことができる。この 12 文字の表記で，「computer」と「computers」の合計 17 文字をそのまま入力しているのと同じ状態になり， 5 文字分の入力を減らし，かつ， 1 回の検索で両方をまとめて検索することができる（http://www.screencast.com/t/ONAvNvRu）。

まず，検索機能として「Grep」を選択し，検索対象とするファイルやフォルダの設定を行う（参照：3.1.1）。次に，検索する表現として，

```
computer[s]?
```

と入力し [2]，正規表現が機能するように，チェックボックスにチェックが入っているかどうかを確認した上で，「検索」のボタンを押す（参照：図 9）。

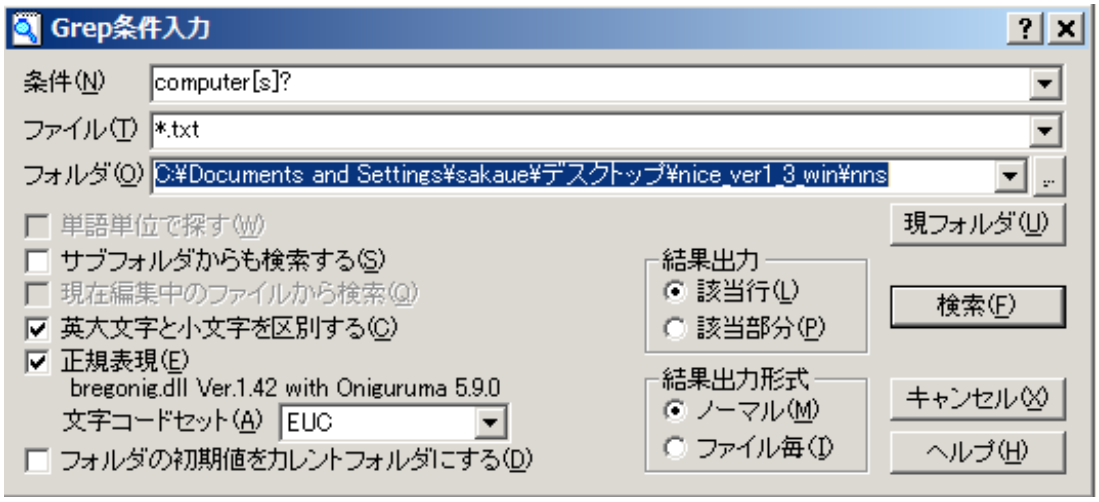

図 9.　computer と computers を同時に検索する条件を入力した画面

検索結果を見ると， 1 回の検索で，「computer」と「computers」の両方が検索され，合計で 16 件マッチする。このように正規表現を使うことで，効率よく検索ができる。

3.3　検索（中級編）

3.3.1　学習者の書いた作文のみを抜き出す

3.1.2 節では，NICE から「computer(s)」という 2 つの単語を同時に検索したが，検索結果には，学習者の書いた文も含まれており，学習者の書いた作文（「*JPN」で始まる行に記録）を母語話者が添削した文（「%NTV」で始まる行に記録。以下，「添削文」と呼

ぶ）も含まれていることがわかる。

学習者コーパスの調査では，「学習者が何を書いているのか」を調べることが中心となるため，NICE のデータを検索する際は，母語話者による添削文は検索対象から外す必要がある。そのため，ある表現を検索する前に，「学習者の書いた文のみを抜き出す」という処理が必要になる。処理の手順は，以下の 2 点となる。

① 学習者の書いた文のみを抜き出す。
② 不要な情報（話者記号などの学習者の文ではない文字列）を削除する。

はじめに，学習者の書いた文のみを抜き出す。学習者の書いた文は，「*JPN」で始まることを手がかりとし，「Grep」検索を行う（http://www.screencast.com/t/fSToWbSvcF）。なお，この時は，正規表現が機能しないように，チェックボックスにチェックを外しておく必要がある。検索の結果，5461 個の文が抜き出される。

次に，不要な情報を削除する。検索結果には，フォルダの名前（「My Documents」など）や「*JPN」という学習者の文であることを識別するために入力した情報（図 11 でハイライトされれいる部分）が含まれており，これらは学習者の書いた文ではなく，削除する必要がある。そのため，サクラエディタの「置換」機能を利用し，「*JPN」を含む検索結果の左側に並んでいる情報を一括で削除する（http://www.screencast.com/t/QLZ02ET4V）。

置換の条件を指定する画面では，「置換前」の文字列として，

```
^.*¥*JPN.*¥t
```

と入力し [3]，正規表現を使って学習者の文ではない箇所（文字列）を指定する。また，「置換後」の文字列には，何も入力しない（参照：図 10）。「ある文字列を何もないものに置換する」ということは，つまり，文字を削除するということになる。なお，置換を実行する前に，「該当行マーク」というボタンを押すことで，どの文字列が置換（今回の場合は，削除）されるのかを確認することができる。サクラエディタの場合，適用範囲は黄色でハイライト表示されるため（参照：図 11），残しておくべき必要な文字列がハイライトされていないかを事前に確認しておくとよい。正規表現による条件指定に問題がなければ，置換を実行し，不要な文字列が削除さされていることを確認する。

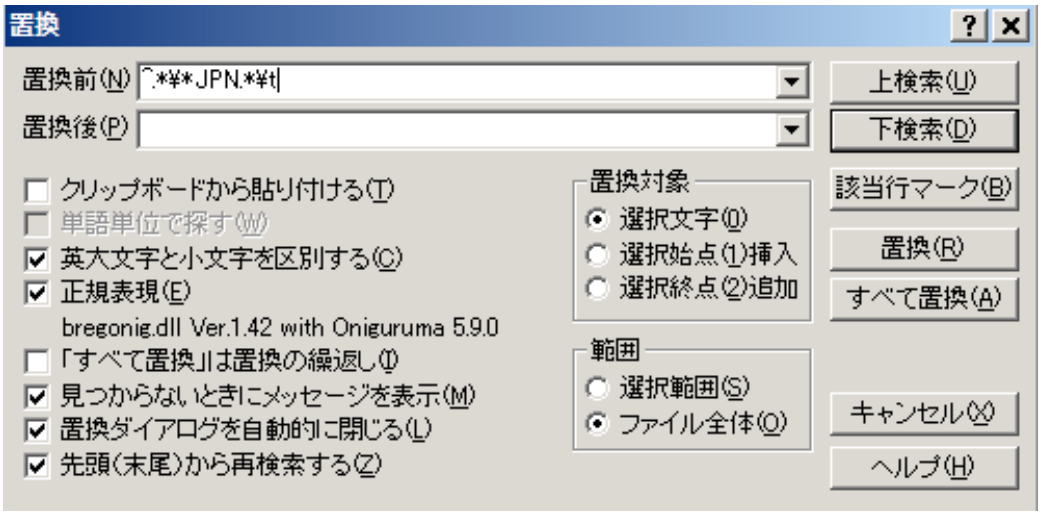

図 10.　不要な情報を削除するために置換の条件指定をした画面

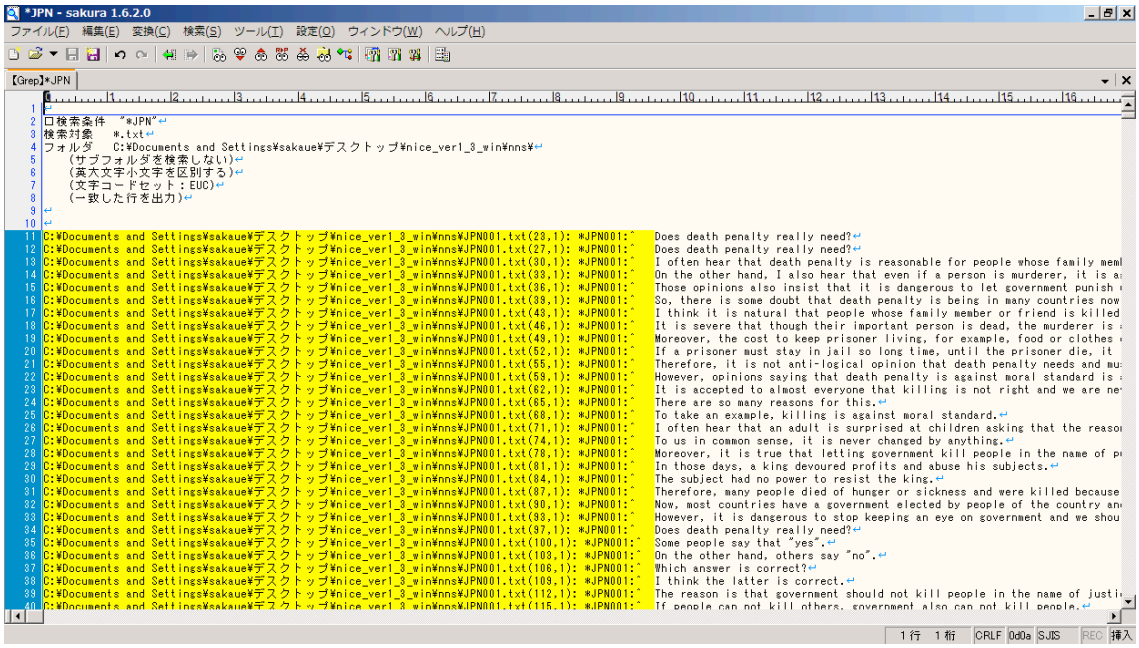

図 11.　条件の適用範囲がハイライトされた画面

3.3.2　TOEIC スコア取得者を見つけ出す

NICE，書き手の属性をもとにしてデータを選別することができる。例えば，英語の能力試験の結果が記録されているため，TOEIC のスコアを基準に，データを上級者と中級者に分けることが可能である。選別のためには，まず，TOEIC スコアの情報を取得する必要がある。実際のデータを図 12 に示す。

```
@Begin
@Participants:      JPN002
@Age:  20
@Sex:  M
@YearInSchool:      U1
@Major:       medicine
@StudyHistory:      8
@OtherLanguage:     Spanish=0.6;none=
@Qualification:     TOEIC=620(2005);none=;none=     #TOEIC スコアの記録有り
@Abroad:      none=;none=
@Reading:     2
@Writing:     1
@Listening:   1
@Speaking:    1
@JapaneseEssay:     3
@EnglishEssay:      3
@Difficulty:  4
@Topic:       school education
@Comments:
@Coder:       2005-12-21 DataInputBy SAKAUE Tatsuya;
@Version:     1.1
*JPN002:      THE JAPANESE SCHOOL EDUCATION
%NTV:  The Japanese education system
%COM:  A more descriptive title would be better. How about something like
"Problems will [worsen/continue] without a return to the old Japanese
education system" or "Current Japanese education system doesn't measure up"?
```

図 12.　NICE のデータの一部

図 12 から，NICE のファイルの冒頭に，行頭がアットマーク（@）で始まっていることが分かる。このアットマークで始まる部分に，書き手の属性が記録されている。ここでは，TOEIC スコアを取得するため，「@Quaification」で始まる行を抽出すればよいということになる。

ここで，サクラエディタの「Grep」機能を使って検索する。「@Quaification」で始まる行は，「TOEIC=700」というように「テスト名＝スコア」という形式で記録されているため，検索条件には「TOEIC」と入力し，検索する。（http://www.screencast.com/t/O5ElGO0l）。検索の結果，121 件がヒットする。

「Grep」機能による検索を行うと，検索条件が含まれた 1 行すべてが結果として表示されるため，この処理の後に，3.2.1 節で行った処理と同様に，不要な情報（行の始めにあるフォルダ名や他の能力テストの情報）を削除し，確認しやすくなるように整形する必要がある。整形されたデータを Excel などの表計算ソフトウェアに読み込ませることで，スコア取得者の一覧表を作成し，取得者の平均スコアを算出したり，高いスコアが記録されたデータはどれかを確認したりすることができる。

検索結果から，フォルダの名前や他のテストの結果など，不要な情報が含まれていることが分かる。そこで，今回は，エディタの「置換」機能を利用して，不要な情報を削除し，必要な情報として，1) 学習者の ID 番号，2) TOEIC というテスト名，そして，3) TOEIC のスコアの 3 つを残すことにする。

「Grep」機能で TOEIC スコアを含む行を抽出した後で，「置換」機能を選択し，置換前の文字列として，

```
.*?(JPN...).*?(TOEIC)=(...).*
```

を入力し，置換語の文字列として

```
¥1¥t¥2¥t¥3
```

を入力する[4]。この置換処理により，不要な情報が削除され，

```
JPN002[TAB]TOEIC[TAB]620
JPN005[TAB]TOEIC[TAB]595
JPN006[TAB]TOEIC[TAB]905
JPN011[TAB]TOEIC[TAB]860
```

というように，1) 学習者の ID，2) TOEIC という文字，3) TOEIC のスコアという 3 つの情報が残されて表示される（http://www.screencast.com/t/ntAIuJHc）。この結果のデータをコピーし，Excel のワークシートへ貼りつけることで，各情報が 3 列になって表示される。もし，１つのセルに３つの情報がまとめてコピーされてしまうなど，コピーが上手くいかない場合は，「テキストファイルウィザード」を利用し，タブを区切り文字とすることで，１つのセルに１つの情報を入れることができる（参照：図 13）。

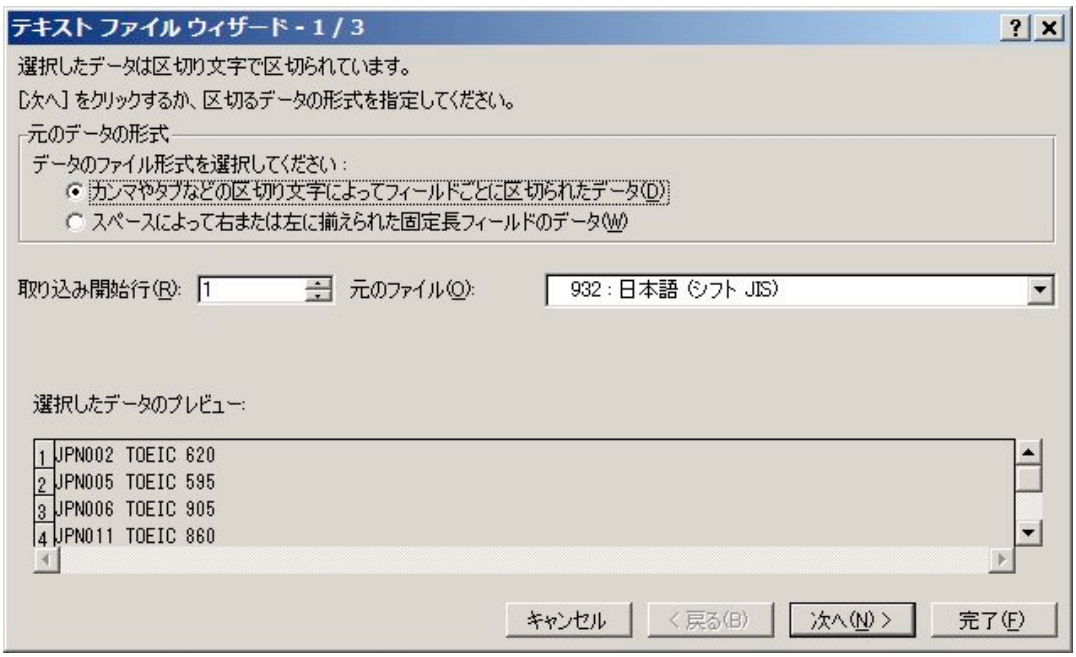

図 13.　テキストファイルウィザードの画面

3.2.1 節での処理と，本節での処理を行うことで，スコア取得者の一覧表を作成することができる。この一覧表を利用し，学習者を習熟度別に分類することができる。また，一覧表から，スコア取得者の平均値や標準偏差なども算出し，「データのばらつき」を観察することで，分析時に習熟度別に学習者を分類できているかどうか，バランスのとれたグループになっていると言えるかどうかを確かめることができる。

3.4 検索（上級編）

基本的な正規表現を覚えたら，以下に紹介する書籍を参照し，自分の関心のある構文等を正規表現で指定できるかを試してみるとよい。

- 宮前 竜也 (2006). 『正規表現ポケットリファレンス』技術評論社.
- 岩谷 宏 (2008). 『入門 正規表現 ~検索・置換・テキスト処理に強くなる!』技術評論社.
- Jeffrey E.F. Friedl (著), 株式会社ロングテール (翻訳), 長尾高弘 (翻訳) (2008). 『詳説 正規表現 第 3 版』オライリージャパン.

3.5 総語数を求める

本節では，NICE に含まれる単語がいくつあるかを数え，どのような単語があるのかを一覧にする方法を説明する。

データの処理にあたり，「サクラエディタ」と「Microsoft Excel」（以下，Excel）を使用する。Excel を使ってのデータ処理は，単語の一覧を作成する際に必要となるが，この内容は次節で詳述する。

単語を数えるためには，まず，学習者の書いた文のみを抜き出す処理が必要である。（参照：3.2.1 節）。学習者の文のみを抜き出した後で，次に，1 行に 1 単語のみが並ぶ形に整形する処理を行う。この整形処理を行い，その処理結果の行数を数えることで，NICE の総単語数（延べ語数）が分かる。

1 行に 1 単語が並ぶようにするために，空白を目印にして，「その空白を改行に置換する」という処理を行う。なお，今回の処理では，説明を簡潔にするため，縮約形（例：don't や isn't など）を 1 語として扱うことにし，ピリオドなどの記号類の処理は行わないものとする。

学習者の文のみが残っている状態（参照：3.2.1 節）で，さらに置換処理を行う。置換処理の前に，データの冒頭と末端にある日本語文は削除しておき，その後，「検索」メニューから「置換」を選択する。「置換前」の欄には半角のスペースを 1 つ入力し，「置換後」の欄には「¥n」と入力する。半角のスペースは，見た目には表示されないが，全角の

スペースを間違って入力しないように注意する必要がある。なお，「¥n」は，改行を表す正規表現である（http://www.screencast.com/t/LVPO2OEhu）。

一連の処理を行い，画面左端に行番号が表示されるが，その最後の行番号を確認すると，「70783」となっていることが分かる。つまり，70783 語が，学習者のデータに含まれている単語の総語数ということになる（参照：図 14）。

図 14.　1 行に 1 単語に整形した結果の画面（最後の行）

3.6　単語頻度一覧表を作成する

本節では，基本操作のまとめとして，NICE に含まれる単語の一覧表を作成する方法を説明する。

データの処理にあたっては，引き続き，サクラエディタと Excel を利用する。今回の動画では，Excel 2007 を使用している。Excel の操作が少し難しいように感じられるかもしれないが，1) 並び替え，2) 関数，3) フィルタリングという 3 つの機能を順番に使うことで処理できる。

まず，3.4 節で行った処理に加えて，記号類の削除をする処理を行う。今回は，ピリオド，コンマ，クエスチョン・マーク，エクスクラメーション・マーク，ダブルクオーテンション・マーク，そして括弧の 6 種類を削除する。サクラエディタで，検索メニューから置換機能を選択し，置換前の欄には，

```
[¥.¥,¥?¥!"¥(¥)]
```

を入力する [5]。今は，削除の処理を行うため，置換後の欄には何も入力しない。正規表現の使用に関するオプションにチェックが入っているかどうかを確認し，「すべて置換」のボタンを押すと，5 種類の記号が削除される。

記号類の削除を行った後，「nice_all.txt」と名前をつけてデータを保存する。次に，このファイルを，Excel を使って開き，1 つのセルに 1 つの単語が入った形で表示されていることを確認する（図 15)。

	A	B
1	Does	
2	death	
3	penalty	
4	really	
5	need	
6	Does	
7	death	
8	penalty	
9	really	
10	need	
11	I	
12	often	
13	hear	
14	that	
15	death	
16	penalty	
17	is	
18	reasonable	
19	for	

図 15.　サクラエディタから Excel に貼り付けた単語一覧の一部

まずは，単語をアルファベット順に並べ替える。Excel のメニューから，「データ」→「並び替え」を選択し，昇順で並び替える。処理が終わると，数字が一番上に並び，皇族の行を見ると，「a」などが見つかる。その後，単語の右隣のセルを選択し，COUNTIF という関数を使って，単語の頻度を求める。例えば，A2 のセルにある単語の頻度を数えるには，B2 のセルに，

```
=COUNTIF(A:A,A2)
```

と入力する（参照：図 16)。この場合，「A の列に A2 に含まれる文字列があったらすべて数えなさい」という命令を意味している。残りのセルに，この関数をコピーすることで，各単語の頻度を一括で求めることができる。

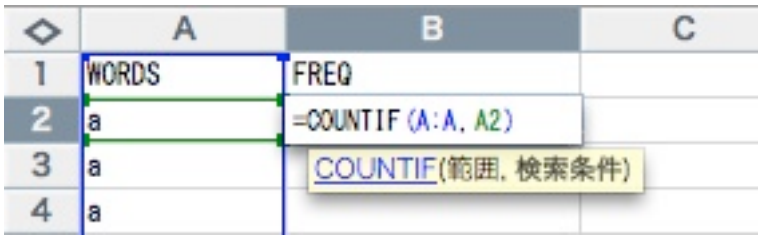

図 16.　countif 関数による単語頻度のカウント

最後に，重複行の非表示を行う。A の列全体を範囲指定した後で，「データ」→「フィルタ」→「詳細設定」という順で機能を選択し，「重複するレコードは無視する」というオプションにチェックが入っていることを確認してから，「OK」ボタンを押す（参照：図 17)。その後，重複している行が消え（正確には，非表示の状態)，各単語とその頻度の一覧が表示される。最後に，この一覧をコピーして，別のワークシートにコピーして貼りつけることで，単語の一覧表が完成する（http://www.screencast.com/t/DcGHsRcw2g)。

図 17.　重複行を非表示にするためのフィルタオプションの設定

本章では，正規表現・テキストエディタ・表計算ソフトを用いての基本的なデータ処理方法について説明した。なお，学習者コーパスのより詳しい処理方法については，「学習者コーパス入門」（http://dictionary.sanseido-publ.co.jp/wp/tag/NICE）を参照されたい。

4.　データ処理・分析に関する注意点

本章では，コーパス分析に関わる注意点を 4 点述べる。

4.1 単語の抽出方法

3.5 節において，単語の頻度一覧表を作成する方法を説明した。3.5 節では，最も単純な方法で文を単語に分けているが，語彙頻度をより正確に求めるには，以下の 2 点を考慮する必要がある。

1) 大文字の小文字への統一
2) 単語のレマ（lemma）化

まず，大文字を小文字に統一することが必要である。同じ単語であっても，コンピュータでは別の単語として扱われてしまう。例えば，文頭にある「The」と，文中にある「the」は別のものとして扱われる。そのため，文頭にある「The」をすべて小文字の「the」に置換する必要がある。サクラエディタでは，置換の機能を使い，置換前の条件として「`[A-Z]`」，置換後の条件として「`[a-z]`」と指定して置換をすると，大文字が小文字に変換される[6]。

次に，単語のレマ（lemma）化をする必要もある。レマというのは，「見出し語」という意味である。例えば，「do」は，「did」で出現することもあれば，「done」で出現することもあるため，これらを別の単語として扱うのか，元は「do」であるので， 1 つの単語としてまとめて扱うのかを考える必要がある。しかし，実際のところ，レマ化する処理は，特殊なプログラムを実行させる必要があり，研究上不可欠な作業でない限りは，レマ化をしないということもある。

4.2 意味の区別

英単語の中には，同じスペルでも，異なる意味を持ったものがある。単純な例として，「can」が挙げられる。「助動詞」としての can はもちろんのこと，「缶詰」という意味で名詞としての can も存在する。このように，同じスペルでも異なった意味を持つ単語がコーパス中に含まれる可能性がある。研究の目的により，こうした単語の意味を区別する必要があるが，その方法は，「品詞付与を行い，自分の目で確かめる」ことである。品詞付与を行うことで，助動詞と名詞という品詞レベルでの区別ができるようになる。しかし，プログラムを使って機械的に品詞付与を行っても，本当に正しい品詞が付与されているかどうかの確認は，最終的に，人の目で行う必要がある。

4.3 処理過程を明らかに

コーパス分析では，テキスト処理という過程を避けることは難しく，自らの研究目的に応じた整形や置換といった処理が必要になる。しかし，世の中には，コーパス検索用ツールがインターネット上で入手・利用することができるため，こうした処理に不慣れな場

合，その過程を飛ばしてしまうことがある。その結果，不要な情報を削除しなかったり，行またぎの表現の検索漏れが生じたりするなど，処理に関する問題を残したまま分析結果が示され，最終的に，誤った結果を導くことになりかねない。

研究上重要なことは，「処理過程をできる限り明らかにすること」である。処理過程が明らかになってさえいえば，仮に間違った処理をしていた場合でも，気づいた者がその問題点を指摘し，解決に至ることができる。しかし，その過程を明らかにしない，あるいは，検索用ツールによって，処理過程を「ブラックボックス化」させた場合，その処理に問題があるかどうかが判断できず，研究結果そのものの信頼性が揺らぐことにもなる。また，過程が明らかにされなければ，調査結果の再現性も失われてしまうため，その点でも結果の信頼性が揺らいでしまうことになる。したがって，どのような処理過程を経て，どのような結果が導かれたのか，その道筋を極力明らかにすべきである。

4.4 身につけるべき知識と技量

コーパスデータの分析に必要な知識として，a) 書かれた内容を理解するための言語知識と，b) データ処理に関する知識の 2 つが必要である。

まずは，そのコーパスデータの分析は，テキスト処理と頻度などの数値データの取り出しに苦心し，データそのものをよく読まずに，数値のみで結果を論じてしまうことがある。どのような文脈で，どのような意味を表そうとして学習者が産出をしているのかについては，実際の英文をよく読むことが必要である。

また，コーパスデータを扱う以上，データ処理に関する一定の知識・技量は必要不可欠である。前節でも述べたように，検索ソフトウェアに過度に依存し，自分が何をどのように処理しているかを全く説明できないようなデータは，分析に使用すべきではない。一定の知識・技量として，具体的には，「正規表現による整形および置換」は必須のものと言える。言い方を変えれば，その技量を身につけることで，NICE に限らず，どのようなコーパスデータでも，自分の研究に適した形に処理ができるようになる。

なお，コーパスデータの分析において，「統計処理」に関する知識も必須のものとなりつつあるが，上記 2 つの知識を身につけることが先決だと言える。また，統計処理を行うと，何もかもが客観的に示され，信頼のできる結果を出せる，有意な差が認められなければ調査として失敗であるという誤った認識がなされていることもあるようだが，安易に統計処理をするのではなく，まずは，どのような過程を経て分析を行い，言語事実としてどうなっているのかを示すことに注力すべきである。

5. おわりに

コーパスを扱うことができれば，客観的なデータに基づいた研究が効率よくできると思われることがある。しかし，コーパスを扱うことで，むしろ，分析に関わる作業量は増

えており，慎重を期して処理を進めなければ，誤った結果を導くことさえある。誤った結果を導いてしまった場合でも，自分の処理手順を振り返り，問題点に気づくことができればよいが，その問題点に気づけず，先へと進んでしまうことは大変危険である。特に，現在は，便利なコーパス検索用のソフトウェアも配布されており，ソフトウェアに過度に依存することが原因で，データ処理上の問題点に全く気づかないことさえある[7]。

本稿では，基本的なデータの処理方法を説明したが，基本的な処理技術が身につけば，NICE だけでなく，他のコーパス処理にも応用できるようになり，データそのものの問題点や処理に関する注意点に気づくことができるようになるはずである。本稿が，研究活動の一助となれば幸いである。

謝辞

本稿の執筆にあたっては，名古屋大学の杉浦正利先生と同志社大学の北尾謙治先生から貴重なコメントをいただいた。末筆であるが御礼を申し上げたい。

注

1. 平成 17, 18, 19 年度科学研究費補助金 基盤研究(B)「英語学習者のコロケーション知識に関する基礎的研究」（代表：杉浦正利（課題番号:17320084））の成果の一部である。
2. 四角括弧（[]）を使うことで，括弧に囲まれた任意の 1 文字にマッチさせることができる。半角の「?」は、直前の文字が 0 回または 1 回だけの出現を意味する正規表現であり，この場合，複数形の s がある場合とない場合を表現することになる。
3. 「^」は行頭を表し，ピリオドは任意の 1 文字を示し，アスタリスクは，直前の文字の 0 回以上の繰り返しを意味する正規表現である。また，半角の「¥」は、正規表現としての特殊な意味を打ち消すために使われる記号である。ここでは，アスタリスクをアスタリスクそのものとして検索するために，正規表現としてのアスタリスクの意味を打ち消す必要があり，そのために「¥」を使う。
4. 丸括弧を使うことで，文字列をグループ化することができる。この場合，JPN と後続する 3 つの数字，また，TOEIC という文字列，スコアの 3 桁の数字をそれぞれグループ化している。グループ化された文字列は，「¥」と数字を組み合わせて参照することができる。例えば，「¥1」と書くと，1 番目にグループ化された文字列にマッチさせることができる。
5. ピリオドは，正規表現のメタキャラクタとして任意の 1 文字を意味するため，そのままピリオドを入力して置換をしても，英文中のピリオドにマッチさせることはできない。そのため，「¥」を直前に書き，メタキャラクタの意味を打ち消して，ピリオドという記号そのものを意味するように指定する必要がある。
6. 範囲を表すハイフンを組み合わせて，`[A-Z]`と書くことで，A から Z までの 26 個すべての英文字を置換の対象として指定することができる。`[a-z]`と小文字で書いた場合は，小文字の a から z までの 26 文字すべてが置換の対象となる。
7. ソフトウェアの利用そのものを否定するものではない。

参考文献

Granger, S., Hung, J., & Petch-Tyson, S. (Eds.). (2002). *Computer learner corpora, second language acquisition and foreign language teaching*. Amsterdam: John Benjamins.

和泉絵美・内元清貴・井佐原均 (2004).『日本人 1200 人の英語スピーキングコーパス』アルク.

古泉隆 (2008). 「学習者コーパスの探索的誤用分析記述方法の概要：NICT JLE コーパスと NICE のデータを使って」杉浦正利（代表）『自然言語処理技術を応用した英語学習者の誤用に関する包括的かつ体系的分析』(pp.137–145). 平成 16，17，18 年度 科学研究費補助金（萌芽研究）研究成果報告書，研究課題番号 16652044

Nesselhauf, N. (2004). Learner corpora and their potential for language teaching. In J. M. Sinclair (Ed.), *How to use corpora in language teaching* (pp.125-152). Amsterdam: John Benjamins.

Sinclair, J. M. (Ed.). (2004). *How to use corpora in language teaching*. Amsterdam: John Benjamins.

投野由紀夫 (編) (2007). 『日本人中学生一万人の英語コーパス JEFLL Corpus —中学生が書く英文の実態とその分析—』 小学館.