2006－NL－172（6）
2006／3／27

対訳辞書のグラフ表現を用いた日英対訳テキストの発見

福島 健一　田浦 健次朗　近山隆
東京大学

対訳コーパスは多言語自然言語処理のさまざまな分野で利用される貴重な研究資源である。対訳コーパス利用上の大きなネックの一つである入手困難性を解決するために、Web から対訳テキストを探し集めようというアプローチが提案されている。本稿は、対訳辞書の取り扱いを工夫し、テキストを前もって別の形式に変換しておくことによって、任意のテキストペアが対訳であるか否かを高速に判定する手法を提案する。この手法により、既存の手法と同等の対訳判定の精度 (F_1 = 0.982) と汎用性を維持しながら、プログラムの実行速度の面で大きな改善を実現した (毎秒 25 万ペア)。また、我々の知る限り汎用的な対訳テキスト収集手法が日本語に適用されたことはなく、本研究が初めての試みとなる。

Detection of Japanese-English parallel text with a graph representation of bilingual dictionary

Ken'ichi Fukushima　Kenjiro Taura　Takashi Chikayama
University of Tokyo

Parallel corpus is a valuable resource used in various fields of multilingual natural language processing. One of the most significant problems in using parallel corpora is the lack of their availability and researchers have tried to solve this problem by collecting parallel texts from the Web. This paper propose a method that judges rapidly whether a pair of texts is parallel or not by devising means of utilizing bilingual dictionaries and processing the texts into another form in advance. With this method, we achieved a significant improvement of execution speed over a similar effort in the past (250,000 pairs/sec), while retaining accuracy (F_1 = 0.982) and generality. In addition, to the best of our knowledge, this is the first proposed method to extract Japanese–English parallel texts from a large corpora.

1　はじめに

対訳テキストは同一の意味内容について複数の言語で記述された、互いに翻訳になっているテキストのペアである。対訳テキストを集積して利用可能な形式に編集したのものを対訳コーパスと言う。統計的機械翻訳の翻訳モデルを学習させたり [1]、対訳辞書やシソーラスという形式で情報を抽出する [9] など、多言語自然言語処理の様々な分野において対訳コーパスが欠かせない。しかしながら、必要な量・種類の対訳コーパスを入手することは一般に困難である。そもそもあまり多くの種類が存在しない上に、それらも言語は英語–フランス語、テキストのジャンルは政府機関の公式文書やソフトウェアマニュアルというように偏っていて、ニーズにマッチした対訳コーパスが見つかることは少ない。

この問題を解決するために Web から対訳テキストを収集するアプローチが提案されている。Web では英語が主流ながらも多彩な言語が使われているし、テキストが扱う話題のバラエティも人間のあらゆる活動のそれに準じると考えられ、対訳コーパス利用上の制約を解決してくれるのではないかと期待できる。

これまでの対訳テキスト収集手法はある程度のサイズの対訳コーパスの生成に成功しているものの、その多くが Web ページの URL 文字列や HTML 構造などの表層的な情報に頼っている。これらはテキストではなくあくまでも Web ページに固有の性質であり、多くの潜在的な対訳テキストが見逃されている可能性がある。一方でそのような表層的な情報に頼らずテキストの意味のみに基づいて対訳判定を行おうとすると計算量が多くなるという問題点がある。

汎用性と処理速度という相反する目的を両立させるために、効率的な対訳判定アルゴリズムを考案した。このアルゴリズムはテキストをあらかじめ整数列に置き換えておくことでテキスト長に比例する時間で対訳の判定を行える。

また我々の知る限り汎用的な対訳テキスト収集手法が日本語を対象に実験・評価されたことはなく、本研究が初めての試みとなる。

2 関連研究

対訳テキスト収集システムについてはいくつもの既存の研究が存在するが、広大な Web 全体から未知の対訳テキストを発見できるものはあまり多くない。ここではそのうち対照的な 2 手法を紹介する。

Ma らは英語–ドイツ語の組み合わせで対訳テキストを収集した [4]。`.de` などのドイツ語圏のドメインに属する Web サイトのリストを作成し、その中のサイト単位で対訳ペアの探索を行う。英独のバイリンガルであるかを調べるために試験的にサイトの一部分をダウンロードし、その時点でバイリンガルであることが確認されればサイト全体をダウンロードする。集めた Web ページの中のすべての英–独ペアについて、対訳か否かの判断を下す。比較対象の 2 つのテキストに出現するすべての単語の組み合わせについて、対訳辞書を参照してその単語ペアが訳語として対応しうるかを調べる。対訳辞書中にその組み合わせが見つかれば対訳らしさを支持するものとして数え上げ、最終的に辞書中に見つけられた単語ペア数のテキスト長に対する比をもって対訳らしさの評価値とする。この値があらかじめ設定された閾値を上回ればそのペアは対訳であると判断される。3145 サイトを 10 台の計算機で 20 日間かけて処理して約 63MB の対訳コーパスを獲得している。

このように単純にすべてのペアについて判定を行うと、その計算量はテキスト数の 2 乗に比例する大きなものになってしまう。実はほとんどの研究では効率を重視して詳細な判定を行う候補ペアの数を一部の対訳 Web ページに固有な条件であらかじめ絞り込んでいる。Resnik らは Web ページペアの URL がある条件を満たしているときにそれらが対訳である可能性が非常に高いことに着目し、そのようなペアに限定して収集を行った [6]。例えば日本語なら `j`, `jp`, `jpn`, `n`, `euc`, `sjis` など、URL には言語や文字コードを示唆する文字列が含まれることがある。この部分文字列を除いて URL が完全に一致するようなページペアのみからなる候補ペア集合を抽出し、それらについてのみテキストの内容に踏み込んだ詳細な判定を行う。8294 の候補ペアの中から 2190 の対訳ペアを発見している。ただこの URL 条件は非常にきつく、20T バイトのデータを処理しながら 8294 ペアしか対訳ペアの候補と見なしていない。

3 提案手法

対訳判定の手がかりとなる情報には URL 文字列や HTML 構造、サイトのディレクトリ構

造などの Web ページに固有なものと、テキストそれ自体がもつ言語情報がある。前節で説明したように、Web ページに固有の性質を利用すれば高い効率で対訳テキストを見つけられるが、Web 上の対訳テキスト全体から見ればそのような性質を満たすものは少ない。一方でテキストの内容だけに頼って判定を行おうとすると判定の回数がテキスト数の 2 乗になってしまう。

ただし、1 回の判定にかかる時間を軽減できればそれだけ多くの候補ペアを同じ時間で処理できるようになる。本研究では辞書の取り扱い方法を工夫し、テキストをあらかじめ数列に変換しておくことで個々の判定を高速に行おうと試みた。

3.1 対訳辞書のグラフ表現

テキストを数列に置き換えるためには、単語から数値への変換規則が必要である。我々は対訳辞書をグラフとして表現することによってこの変換規則を作り出した。まず対訳辞書に登場する各単語をノードと考える。次に辞書のエントリを一つずつ見ていき、そのエントリが結びつける 2 単語に対応するノードをエッジで結ぶ。こうして出来上がったグラフは複数の連結成分からなるが、それぞれが一つの「意味」を表現する単語の集合だと解釈し、各連結成分に「意味 ID」というユニークな番号をふる。単語文字列からその単語が属する連結成分の意味 ID への置き換えが変換規則である。

3.2 テキストの前処理

テキストを単語レベルに分解し、各単語を上で説明した変換規則によって数値に置き換える。辞書にない単語は無視する。辞書が持つ対訳情報の他に有用かつ扱いやすい情報として、単語のテキストにおける出現位置がある。例えば、辞書によれば対訳でありうる単語ペアでも一方はテキストの先頭に出現し他方は末尾に出現する場合、これを対訳らしさを支持するものとして数え上げることは判定の精度に悪影響を与える。この情報も活用するために、テキストの長さを 1 に正規化してそのときの座標を出現位置とし、各単語に持たせる。結局、各単語は (意味 ID, 座標) という 2 次元ベクトルで表現される。最後に、この 2 次元ベクトルを意味 ID、座標の順でソートしてテキストの数列表現が完成する。この処理はテキスト長 n に対しオーダー $n \log n$ の時間を要するが、各テキストについて 1 回だけ行えばよい。

3.3 対訳らしさの計算

数列表現に置き換わった 2 つのテキストの比較は、次のようにして行う。まず両方の配列の先頭にカーソルをセットする。もし 2 つの要素の意味 ID が一致し、なおかつ座標の差、すなわち距離があらかじめ設定した閾値より小さければこの組み合わせを対訳な単語ペアとしてカウントし、両方のカーソルを一つ先へ進める。そうでない場合は、数列のソートと同じ基準で小さいほうのカーソルだけを進める。いずれかのカーソルが数列の終端に達するまでこの操作を繰り返し、見つかった対訳な単語ペア数の数列長の和に対する比をもって対訳らしさの評価値とする。1 回の操作で必ず 1 つのカーソルは先に進むので、このアルゴリズムは高々 2 つの数列の長さの和に比例する時間内に終わる。擬似コードは図 1 のようになる。

3.4 対訳グラフの分割

テキストを数列で表現することで効率的に対訳の判断を行えるようになったが、実は対訳辞書からつくったグラフをそのまま変換規則として使うのには大きな問題がある。対訳グラフにはいくつものエッジを介して意味的に何の関連もない単語同士を結び付けてしまうという欠点がある (図 2)。そのような偽りの対訳関係も数列表現においては区別せずに扱われ、判定の品

```
/* 数列の 1 要素を表現する構造体 */
typedef struct {
  int notion_id;
  double coord;
} element;

/* 要素の比較関数 */
int
less(element e1, element e2)
{
  if(e1.notion_id == e2.notion_id){
    return e1.coord < e2.coord;
  }
  return e1.notion_id < e2.notion_id;
}

/* 対訳らしさを評価する関数 */
double
calc_score(element *text1, size_t sz1,
           element *text2, size_t sz2,
           double threshold)
{
  int count = 0;
  size_t pos1 = 0, pos2 = 0;
  while(pos1 != sz1 && pos2 != sz2){
    if(text1[pos1].notion_id
         == text2[pos2].notion_id
      && fabs(text1[pos1].coord
         - text2[pos2].coord)
         < threshold){
      count++;
      pos1++;
      pos2++;
    }
    else{
      less(text1[pos1], text[pos2])
        ? pos1++ : pos2++;
    }
  }
  return (double)count / (sz1 + sz2);
}
```

図 1　数列表現によるテキスト比較のコード例

質を傷つける。

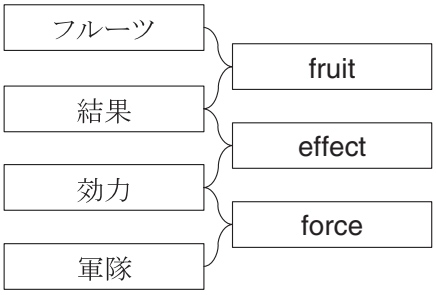

図 2　無関係な語を結びつける対訳グラフの例

いわゆる多義語はその訳語も多義語であることが多く、いくつもの多義語がつながって巨大な連結成分を構成してしまうことがありうる。(今回の実験で扱った辞書では、全 19413 エッジのうち 3943 を最大の連結成分が含んでいた。) 偽りの対訳関係の数は単純にはノード数の 2 乗に対応するので、巨大な連結成分をそのまま変換規則として利用すると判定精度を大きく損なう可能性がある。したがって巨大な連結成分は適切なサイズまで分割を行う必要がある。

4　実験と評価

4.1　準備

本研究ではアルゴリズムの性能を評価するために、対訳辞書は EDR 電子化辞書 [2] を、サンプルデータは Fry が集めた日英対訳 Web コーパス [3] を利用して実験を行った。今回は名詞のみを考慮に入れて対訳辞書を編集した結果、ノード数 28001、エッジ数 19413 の対訳グラフを得た。このグラフは 9178 個の連結成分からなり、最大の連結成分は 3563 のノードと 3943 のエッジからなる。これを含めた大きな連結成分は分割する必要があるが、条件を変えていろいろな大きさで分割を行った。さらに、分割の際に切断されてしまったエッジの情報も補完して利用したり、簡単に辞書を増強する方法として数詞の追加を行うなど、同じ分割に対しても

バリエーション用意し、480 種の変換規則を作成した。

コーパスからランダムに選び出して実際に対訳であることを手作業で確認した 408 ページペアのみを使い、テキストの単語への切り分け・品詞タグ付けには日本語は和布蕪 [5] を、英語は SS Tagger[7] を利用した。さらに英語の活用形を原形に直すために WordNet[8] の `morphstr()` を利用した。

4.2 実験結果

まず巨大な連結成分が判定精度に与える影響とそれが分割によってどう改善されるのかを図 3 に示す。これらは上で説明した実験用データの全組み合わせ $408 \times 408 = 166464$ ペアから正解の 408 ペアをどれだけの精度で見つけ出せるかという数値である。分類アルゴリズムの一般的な評価指標である F_1 値を用い、分類のスレッショルドを段階的に変化させて最大の F_1 を得たときの、その値を記している。この時点ではまだエッジの補完や数詞の追加は行っていない。最も大きな連結成分をそのまま扱った場合 (a) と無視した場合 (b) を比較することで、巨大な連結成分が判定アルゴリズムを激しく撹乱することがわかる。さらに分割を行ってそのような大きな連結成分が持つ情報も活用することで精度が上がっている (c)。各欄の上/下段のグラフは単語間距離にスレッショルドを設定する/しない場合を表している。どの条件においても距離の条件を設けることでより良い精度を得ている。

今回用意した 480 種の辞書のうち、最も良い精度を示したのは [日英で少ないほうのノード数が 10 以下になるまで分割する、切断されたエッジの情報も利用する、0〜9999 の数詞を追加する] というものであった。さらに距離のスレッショルドを変えながら実験を行ったところ、$\max F_1 = 0.982$ の精度を得た。このときの precision–recall 曲線を図 4 に示す。

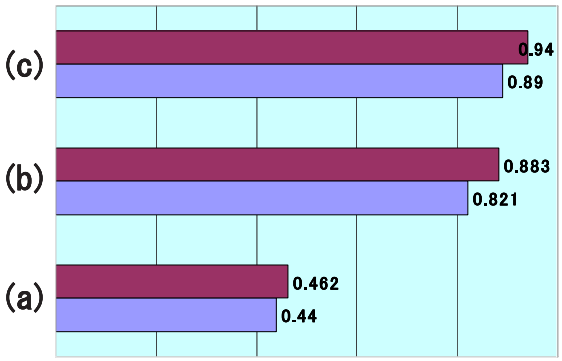

図 3　グラフ分割の効果

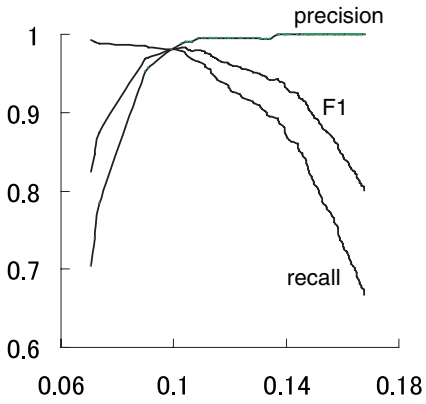

図 4　最良の精度を示した変換規則

対象とする言語をはじめとして様々な条件が異なっているので Resnik らや Ma らによる既存研究との比較には慎重になる必要があるが、Resnik らが正例混入率 134/149 のデータで $F_1 = 0.969$、Ma らが 240/300 で $F_1 = 0.981$ だったのに対し、我々ははるかに正例混入率の低いデータで同等の数値を得ている。

4.3 大規模 Web データの処理にむけて

本研究の最終的な目標は Web から未知の対訳テキストを発見することにある。実験用の小さなデータでは本手法が高い精度で対訳判定を行えることを示したが、現実の Web にはそのような高い頻度では対訳テキストは存在しないと考えられる。そこで実験用データの正例混

入率を大幅に下げて実験を行った。408 の正例に無関係な Web ページを混ぜ、正例の混入率が 408/約 1000 万というデータを作った。前節の最後と全く同じ条件で実験を行ったところ、$\max F_1 = 0.931$ という数値が得られた。このとき precision は 0.978 を保っており十分実用に耐えうるものと考える。

また本手法の強みとして主張している実行速度であるが、こちらは 1CPU あたり毎秒平均 25 万ペアという数値を得た。使用したプロセッサは Xeon(2.4GHz) である。Ma らの論文には実験の条件が詳しく記述されておらず、比較のためにはいくつもの仮定を持ち出さざるを得ないが、仮に一つの Web サイトに各言語のページが 1000 ずつ、つまり 10^6 の候補ペアが存在し、7 年間で計算機の処理性能が 32 倍になったとすれば、本手法は Ma らのそれの 40 倍超の速さで対訳判定を行える。この差を生み出した原因はアルゴリズムのオーダーではないかと思われる。論文を読む限りでは、Ma らは比較対照の 2 つのテキストに出現する単語の組み合わせを列挙し、そのすべてについて辞書を参照しながら訳語として対応しうるかを調べることによって対訳らしさの評価値を計算している。この方法ではテキスト長の 2 乗と辞書の検索コストに比例する時間がかかる。一方、我々が提案する手法ではテキスト長に比例する時間で 1 回の判定を行える。

5 おわりに

本稿では、辞書をグラフ形式で取り扱い、テキストをあらかじめ別の形式に置き換えておくことによって、任意のテキストペアが対訳であるか否かを高速に判断する手法について述べた。この手法は精度、処理速度、汎用性を高いレベルで兼ね備える。精度は既存の手法と同等以上の $F_1 = 0.982$、速度は毎秒 25 万ペアという実験結果を得た。またテキスト自体が持つ言語情報しか使っていないので、Web 上のどんな対訳ページペアも発見可能である。

今後は実際に本手法を運用して、大量の Web データからの未知の対訳テキストの発見を試みる予定である。

参考文献

[1] Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. A statistical approach to machine translation. *Comput. Linguist.*, Vol. 16, No. 2, pp. 79–85, 1990.

[2] EDR 電子化辞書. `http://www2.nict.go.jp/kk/e416/EDR/J_index.html`.

[3] John Fry. Assembling a parallel corpus from RSS news feeds. In *Workshop on Example-Based Machine Translation, MT Summit X, Phuket, Thailand*, September 2005.

[4] Xiaoyi Ma and Mark Liberman. Bits: A method for bilingual text search over the web. In *Machine Translation Summit VII*, September 1999.

[5] Mecab: Yet Another Part-of-Speech and Morphological Analyzer. `http://chasen.org/~taku/software/mecab/`.

[6] Philip Resnik and Noah A. Smith. The web as a parallel corpus. *Comput. Linguist.*, Vol. 29, No. 3, pp. 349–380, 2003.

[7] SS Tagger - a part-of-speech tagger for English. `http://www-tsujii.is.s.u-tokyo.ac.jp/~tsuruoka/postagger/`.

[8] WordNet. `http://wordnet.princeton.edu/`.

[9] 長尾真（編）. 自然言語処理. 岩波ソフトウェア工学, No. 15. 岩波書店.