HMMに基づく多言語固有表現抽出システムの開発

齋藤 邦子　　永田 昌明
日本電信電話株式会社　NTTサイバースペース研究所

1　はじめに

インターネットの普及が進む現在、ネットワークを通じて様々な言語で書かれた情報に接する機会が日々増加している。ある検索エンジンの2000年の調査では、全世界のWebページの分布は、1位：英語(76.6%)、2位：日本語(2.77%)、3位：ドイツ語(2.28%)、以下、中国語(1.69%)、フランス語(1.09%)、スペイン語(0.81%)、韓国語(0.65%)と続いている[1]。分布の大半を占めている英語は勿論のこと、日本・中国・韓国などのアジア圏からも有益な情報を得られなければ、折角の豊富な情報資源を十分活用しているとは言えない。

そのため近年は日本語だけではなく外国語、特に英語やアジア系言語からも情報収集し、翻訳して内容を理解したいという要望は非常に高い。このような多言語情報資源を活用するためには、扱いたい言語についての解析技術の開発が必須となる。なかでも形態素解析・固有表現抽出技術はテキストの単語認定・内容理解のための基盤技術であり、検索・要約・翻訳などの様々な自然言語処理アプリケーションに利用されている。

そこで我々はHMMに基づく多言語固有表現抽出システムを開発した。本システムは、主に英日中韓を対象とし、共通の学習・解析エンジンと言語別辞書の差し替えにより多言語化を実現した。本稿ではシステム概要および固有表現抽出で用いるHMMについて報告する。

2　システム概要

本システムの概要を図1に示す。本システムは主に英日中韓を対象とするが、原理上は任意の言語を扱えるので処理対象となる言語をX語とする。

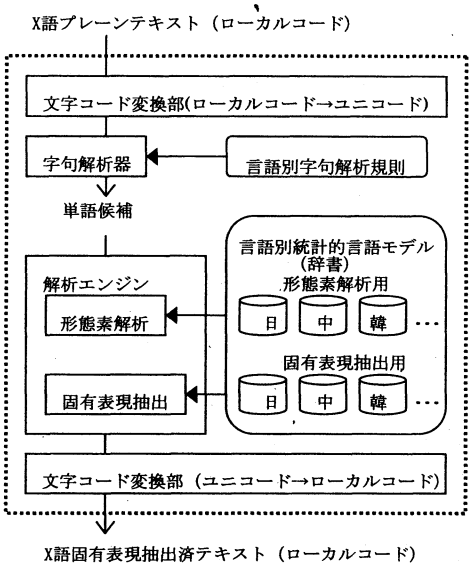

図1.　システム概要図

以下、順にシステムの構成要素である文字コード変換部・字句解析器・解析エンジンを説明する。

2.1　文字コード変換部

文字コード変換部はテキストの文字コードをローカルコードとユニコード間で変換する。システム入力時にローカルコードからユニコードへ、出力時にユニコードからローカルコードへと変換し、システム内では全言語をユニコードで処理する。

ローカルコードとは計算機で文字を扱うために言語別に設定されているコードセットであり、例えば日本語ではEUC-JP・SJISなど、中国語ではGBなど、韓国語ではKSCなどがある。これらのローカルコードは異なる言語を同時に扱うことができないため、世界中の言語を1つのコードセットにまとめたものがユニコードである。ユニコードを用いると英語・日本語・中国語などを同時に扱う

ことができ、複数の言語を処理する多言語解析技術には不可欠なものである。ユニコードではアルファベット・数字・記号・漢字（日中韓共通）・ひらがな・カタカナ・ハングルなどの文字種のコードポイント範囲がプロパティとして定義されているだけでなく、ユーザが目的に応じてプロパティを自由に定義することも可能である。本システムではプロパティ情報を2.2字句解析器で利用する。

2.2 字句解析器

字句解析とは入力された文字列から単語を切り出す処理のことである。英語などのヨーロッパ系言語は空白によって単語を分かち書きするが、日本語・中国語・韓国語などのアジア系言語の多くは単語を繋げて書く習慣がある。韓国語では空白を用いて途中を区切っているが、単語単位よりも長い文節単位で区切り、区切り方にも個人差がある。そのためアジア系言語ではまず文から単語認定を行うことが不可欠である。

単語を認定するにはまず入力文から単語候補となる文字列を切り出さねばならない。最も単純な手法は各位置における任意の m 文字の文字列を全て単語とみなすものである。これは多くの言語で共通に適応できる手法であるが、候補の中には単語になりえない文字列も大量に含まれる。すると後の解析処理で確率計算の場合の数が膨大となって解析速度が遅くなり、実用上問題がある。そこでより効果的な単語認定の処理が必要となる。

単語の認定には文字種が重要な手がかりとなることが多い。例えば言語共通に言えるのは数字・記号の文字列は製品番号や電話・郵便・番地番号表記であるとか、アルファベット・記号類の特徴的な文字列（http://www.foo..、abc@defg.hi..）は URL やメールアドレスであるということがあげられる。ただし言語によって微妙に流儀が異なる場合がある点には注意が必要である。

言語別に考えると、日本語では文字種の変り目が単語の切目になりやすい。特にカタカナはひとまとまりで外来語になることが多い。また文字種の構成によって平均単語長も異なる。例えば漢字ならば 2 文字前後、ひらがななら 3 文字前後という具合である。中国語や韓国語では文の殆どが漢字またはハングルという同一の文字種で構成されているため日本語ほど文字種の情報が有効ではないが、アルファベット・数字など文字種が変われば単語の切れ目になりやすいという傾向、および文字種によって平均単語長が異なるという性質を利用することができる。例えば中国語では殆どの漢字は 1〜2 文字であるが、外来語を漢字で表現する時は 4 文字程度となる。韓国語では漢字 1 文字がハングル 1 文字に対応し、ハングルは子音-母音-子音を含むので、日本語のカタカナ外来語に相当するものは大体 3 文字程度で表現される。このように文字種に応じた平均単語長の性質をふまえて言語別に字句解析規則を用意し、字句解析器で規則を参照しながら状況に応じた単語候補を生成すれば言語の違いを吸収できる。ここで利用する文字種の情報はユニコードのプロパティから得る。

字句解析器が果たすもう 1 つの重要な役割は言語別に異なる空白の扱いである。日本語・中国語の場合、入力文中の空白は常に単語として認定して出力に含むことが期待される。しかし英語・韓国語のように単語や文節の区切りとして空白を用いる言語の場合は、入力文中の空白を 1 つの単語として認定することは期待されない。例えば

I have a pen →

'I/代名詞' 'have/動詞' 'a/冠詞' 'pen/名詞'

と解析されるべきであり、

'I/代名詞' ' /空白' 'have/動詞' ' /空白'

'a/冠詞' ' /空白' 'pen/名詞'

とはならない。ただし、英語・韓国語では空白を含む単語（複数の単語からなる複合語）は数多く

存在し、この場合は空白を含めた単語認定をしなければならない。

以上のような空白の扱いの差は、2.3 解析エンジンで用いる統計的言語モデルにおいて空白をモデルに含むべきかという問題と関係する。日本語・中国語では殆ど空白が登場しないため、空白が登場した事実が 1 つの重大な手がかりとなるが、英語・韓国語のように区切りとして空白を多用する言語では、空白は連接の手がかりとして重要な情報を持たないのである。

このように言語別、または同じ言語でも状況によって異なる空白の扱いの差を吸収する上でも字句解析器を利用する。具体的には、日本語・中国語では常に空白を単語候補として生成させるが、英語・韓国語では空白 1 文字では単語候補としないで無視し、複合語を生成するときには空白を含める、という規則を記述すればよい。

表 1 は字句解析器が参照する字句解析規則の例（日本語・英語）である。文字種によって切り出す単語の長さを決めてある。言語別に文字種の構成や単語長などの特徴が異なるが、それぞれ規則で書き分けることができる。規則では複数の文字種から構成される単語候補を生成させることも可能で、日本語での漢字かな混じりの単語、英語での複合語などに対応できる。なお、カタカナ・アルファベット・数字での「文字種が変わるまでまとめて切る」では、同じ文字種でひとまとめにし、更にその途中位置からは単語候補を生成させない。例えば「NTT が」であれば「NTT」「が」のみ生成し、途中の「TT」「T」は生成しない。これは字句解析をする時に次の文字位置を 1 文字先にするか、現在生成した単語の終了位置までスキップするかを制御して行う。また数字で小数点や位取りの記号「.」「,」を数字とまとめておきたい場合は、ユニコードの文字種プロパティをユーザ定義し、数字「0〜9」に記号「,.」も含むようにしておけばよい。

表 1. 字句解析規則の例

	文字種	字句解析規則
日本語	漢字	文字種が変わるまで 1-3 字で切る
	ひらがな	文字種が変わるまで 1-3 字で切る
	カタカナ	文字種が変わるまでまとめて切る
	アルファベット	文字種が変わるまでまとめて切る
	数字	文字種が変わるまでまとめて切る
	記号	1 字で切る
	漢字-ひらがな	3 字までの混合を許す
英語	アルファベット	空白が現れるまでまとめる 複合語は空白で 3 単語まで繋げる
	数字	文字種が変わるまでまとめて切る
	記号	1 字で切るが空白単体は無視する

表 2. 字句解析器の実行例

入力文	NTT サイバースペース研究所は東京から約 50km 離れている。	I live in New York for 20 years.
字句解析器が生成する単語候補	NTT サイバースペース 研 研究 研究所 は 、 東 東京 か から 約 50 km 離 離れ 離れて れ れて れてい て てい ている い いる る 。	'I' 'I live' 'I live in' 'live' 'live in' 'live in New' 'in' 'in New' 'in New York' 'New' 'New York' 'New York' 'York' 'York for' 'for' '20' 'years' '.'

実際の規則は正規表現で記述し、字句解析は正規表現に従って動作するため、この字句解析器は多言語を対象として汎用的に利用できる。

表 2 は、字句解析器が表 1 の規則に基づいて単語候補を生成する時の実行例である。日本語では漢字・ひらがなが 1〜3 文字、記号・数字・アルファベットはひとまとまりになっているほか、3 文字までの漢字かな混じりの候補も生成されている。各位置における任意の m 文字の文字列を全て単語とみなす場合と比べれば、非常に効率よく候補が生成できていることは明らかである。英語では空白 1 文字は単語として無視しながら、空白で区切られた文字列を候補とする。更に複数の単語（3 単語まで）からなる複合語も生成される。

2.3 解析エンジン

解析エンジンでは、字句解析器で生成した単語候補に対して統計的言語モデルに基づいて形態素解析および固有表現抽出を行う。本システムでは形態素解析には単語 bigram モデル、固有表現抽出には HMM を利用した。これらのモデルは予め人手で形態素および固有表現情報が付与された正解コーパスから学習する。モデルには言語依存性がないのでモデルの学習エンジンは言語共通であり、コーパスを用意すれば処理対象を任意の言語に拡張することが可能である。

3 固有表現抽出モデル

固有表現抽出の手法にはサポートベクタマシン（SVM）、最大エントロピー法（ME）など様々あるが、本システムでは英語固有表現抽出で高精度をあげたシステム Nymble で提案された HMM をベースとした[2]。HMM は SVM や ME と比べると一般的に精度はやや劣るが、学習・実行速度が速いという点では実用に適している。この HMM は形態素列 $W = w_1 \cdots w_n$ と固有表現列 $NC = NC_1 \cdots NC_n$ の同時確率 $P(W, NC) = \prod P(w_i, NC_i)$ を次式で計算する。

1. $NC_i \neq NC_{i-1}$ の時

   $P(w_i, NC_i) = P(NC_i \mid NC_{i-1}, w_{i-1}) \times P(w_i \mid NC_i, NC_{i-1})$

2. $NC_i = NC_{i-1}$ 且つ $NC_i = NC_{i+1}$ の時

   $P(w_i, NC_i) = P(w_i \mid w_{i-1}, NC_i)$

3. $NC_i = NC_{i-1}$ 且つ $NC_i \neq NC_{i+1}$ の時

   $P(w_i, NC_i) = P(w_i \mid w_{i-1}, NC_i) \times P(< end > \mid w_i, NC_i)$

$< end >$ はある状態での終端を表す特殊記号である。このモデルでは形態素解析と固有表現抽出を同時に行うので、アジア系言語の固有表現抽出でよく問題となる形態素と固有表現の区切単位が異なる点を解消できる。ただしアジア系言語では形態素解析の単語分割に多義が多いことを考慮し、本システムでは予め形態素解析処理を行って Nbest 形態素列候補を単語グラフで出力し、このグラフに対して固有表現抽出の HMM を適応するという改良を加えた。Nbest 候補の出力には前向き DP 後ろ向き A*アルゴリズムを利用した。これにより形態素解析で下位候補にある形態素も固有表現抽出の処理対象となり、単語分割の多義を吸収できる。また学習データ不足の問題を補うためにモデルの各項をより低次の項で平滑化し、更に単語の出現頻度を品詞の出現頻度で線形補間した。

4 評価

本システムを評価するために、独自に英日中韓の正解データ（新聞記事・Web 文書等、5,000～10,000 文程度）を作成して固有表現抽出を行った。なお固有表現抽出の定義は、英：MUC、日中韓：IREX に準拠した。各言語の解析精度は F 値（%）で、英：88.2、日：81.0、中：84.5、韓：79.9 となった。評価データが独自のものであるため、従来技術と単純な比較はできないが、同じデータを用いて SVM で評価すると、英：84.7、日：57.3、中：89.5、韓：82.1 となったことから、本システムの解析精度は従来技術の水準にほぼ到達したと考えられる。

5 まとめ

HMM に基づく多言語固有表現抽出システムを開発した。本システムでは辞書とシステム内は全てユニコード化した。また空白や文字種などの言語の違いを吸収する字句解析器と、言語に依存しない言語モデルを組み合わせることにより、言語共通の学習・解析エンジンと、言語別字句解析規則および辞書の差し替えによる多言語化を実現した。

6 参考文献

[1] Google: 1.6 Billion Served, Wired December 2000, pp.118-119 (2000)

[2] Bikel, D. M., Schwartz, R. and Weischedel, R. M.: An Algorithm that Learns What's in a Name, Machine Learning, Vol. 34, No. 1-3, pp. 211-231 (1999)