

3 自然言語処理の応用

3.1 機械翻訳

機械翻訳はコンピュータを用いてある言語を他の言語へと変換する処理です。このような機械翻訳の手法としては、テンプレート方式や、トランスファー方式、中間言語方式、用例ベース方式等があります。

テンプレート方式では、定型的な文章に対して、翻訳パターンをあらかじめテンプレート化しておき、変化する部分のみを入れ替えて翻訳を行います。例えば、株式市場の速報では、「売買高は、概算約〇〇株」のような定型的な表現が出てきます。この文を翻訳する場合、あらかじめ、Trading volume was estimated at 〇〇 shares. のようなテンプレートを作成しておきます。そして、株数を表す〇〇の部分だけを入れ替えることで翻訳を行います。比較的精度が高い翻訳を実現することができますが、翻訳できる範囲は限定されます。

トランスファー方式は、解析、変換、生成の3ステップにより翻訳を行う方式です。解析部分では、自然言語処理に基づいて、原言語の形態素解析、構文解析、意味解析、文脈解析といった処理が行われ、その文の意味の理解が行われます。変換部分では、意味・文脈構造の変換、構文構造の変換、単語列の変換といった変換処理が行われます。そして、生成部分では、変換結果に基づき、構文構造が作成され、そこに翻訳された単語が配置されることで目的言語の文が形づくられます。

中間言語方式は、一度、原言語を中間言語に変換し、そこから目的言語への変換を行います。中間言語とは、コンピュータ内でさまざまな言語を表現できるように設計された翻訳用の言語です。

ここで、多数の言語間で相互に翻訳を行う場合に必要となるプログラム数について考えてみましょう。ここでは、日本語、中国語、英語、ドイツ語、フランス語の5つの言語間で、相互に翻訳を行うとします。トランスファー方式の場合、日本語から日本語への変換は必要ないことから、日本語から翻訳しなければならない言語の数は、中国語、英語、ドイツ語、フランス語の4つとなります。同様に考えていくと、5つの言語間で相互に翻訳する場合、 5×4 で20個の変換プログラムが必要になることが分かります。一般に、トランスファー方式では、 n 個の原言語から、自分自身を除いた $(n-1)$ 個の目的言語への翻訳を行うことから必要な変換プログラムは $n(n-1)$ 個となります。

一方、中間言語方式では、 n 個の原言語から中間言語への変換のために n 個、中間言語から目的言語への変換のために n 個の合計 $2n$ 個の変換プログラムを用意すればすべての翻訳を行うことができます。5つの言語間で変換を行う場合を考えると、 2×5 で10個の変換プログラムが必要になります。これは、トランスファー方式で必要となる20よりも少なく、中間言語方式の方が効率的に翻訳処理を行うことができます。中間言語方式では、原言語を一度、中間言語へ変換しなければならないため、1つの言語から別の1つの言語のみに変換するような場合は、むしろ処理が増加します。しかし、多言語間の翻訳を同時に行う場合には効率的な翻訳方式になります。

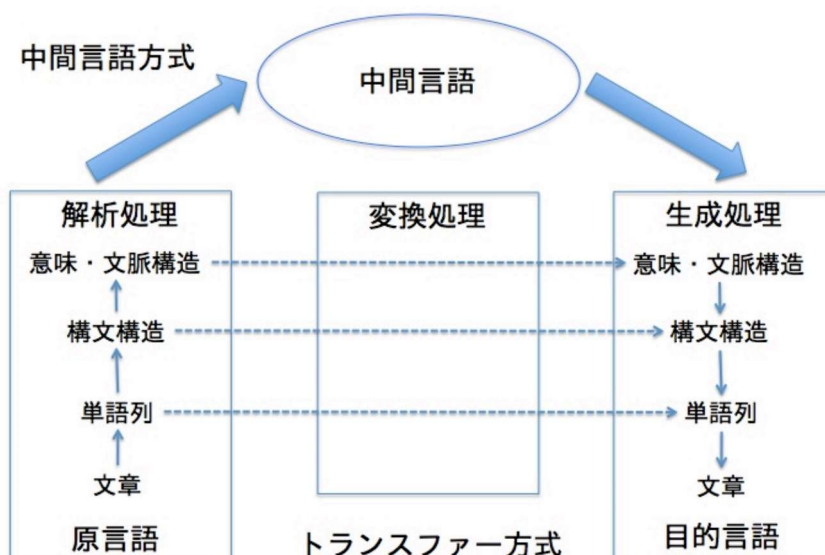


図3-1 ルールベース機械翻訳

トランスファー方式や中間言語方式は、翻訳のための変換ルールを予め用意しておき、それに基づき翻訳を行う方式です。このような翻訳は、ルールベース方式と呼ばれます。これらの方法は、一般性がある反面、例えば、「Time flies like an arrow. (光陰矢の如し)」を「時間 蠅は矢を好む」と訳してしまうというように、対象や状況に応じては正確な翻訳を行えない場合もあります。

このようなルールベース方式に対し用例ベース方式と呼ばれる方法もあります。これは、原言語と目的言語の対訳文を多量に蓄積しておき、それを翻訳に利用するものです。具体的には、対訳文の中から、入力文に類似した用例を探し出し、単語の入れ替え等を行うことで翻訳を行います。また、このような用例ベース方式は、ルールベース方式と組み合わせて、翻訳品質の向上に利用される場合もあります。

機械翻訳を行うためには、自然言語の高度な解析や、変換、生成の技術が必要となります。しかし、意味内容にまで踏み込んだ処理を正確に行うことは、現状では、必ずしも容易ではありません。このため、機械翻訳を利用する場合は、機械が理解しやすいように、もとの文を書き換えてから翻訳を行うといったことも必要になります。

3.2 全文検索

多量に蓄積された文書の中から、目的の検索キーワードが含まれる文書や、その位置を探し出す処理が全文検索です。全文検索を実現するには、いくつかの方法があります。

全文検索を実現するための最も簡単な方法は、総当たりによって検索を行う方法です。この方法では、検索キーワードと一致する文字列を、文書の最初から最後まで順次照合しながら探し出します。このようにして全文検索を実現する場合、あらかじめ前処理を行う必要が無く、任意の文書を対象に検索を行うことができます。しかし、検索対象の文書量が増えれば増えるほど、それに比例して検索時間が増えるという問題があります。

一方、検索処理の高速化のため、あらかじめ検索用の手がかりとなる索引（インデックス）を作成しておく方法もあります。検索用のインデックスとして広く用いられるものの1つとして、転置インデックスがあります。

転置インデックスを作成するには、まず、検索対象となる文書から、形態素解析を行うことでキーワードの抽出を行います。これにより、文書ごとに、例えば、「検索」といったキーワードが含まれているかを知ることができます。逆に、キーワードから、そのキーワードを含む文書へアクセスすることも可能になります。そのためには、図3-2に示すような、単語と、その単語を含む文書IDとの対応表を作成します。このような、単語と、その単語を含む文書IDとの対応表は、転置インデックスと呼ばれます。

例えば、作成した転置インデックスを用いて、「物理」というキーワードを含む文書を検索することを考えてみましょう。まず、転置インデックスの単語の中から「物理」という単語を探し出します。そして、その単語を含む文書IDを見ることで、その単語を含む文書のIDが2と17であることが分かります。より詳細な検索を実現するために、キーワードを含む文書IDとともにキーワードの出現位置を転置ファイルに含める場合もあります。

単語	文書ID
物理	2, 17
情報	1, 17
音楽	5, 9, 14
数学	4, 7, 9
体育	13, 15

図3-2 転置インデックス

このような転置インデックスを用いた検索を更に高速化するために、不要語をインデックスから除外することができます。例えば、英語の文書を検索する場合、「to」や「be」といった、検索の際にあまり重要ではないと考えられる不要語をキーワードから除外します。これにより転置ファイルが小さくなり、無駄な単語同士の照合を減らすことができます。ただし、この場合、「To be or not to be. (生きるべきか死ぬべきか)」といった文を検索できなくなるといった問題もあります。

次に、キーワードの重要度について考えてみましょう。キーワードの重要度を求める方法の1つとして、TF・IDFと呼ばれる指標があります。TFとは、Term Frequencyであり、文書nにおける単語iの出現頻度で定義されます。文書中に頻繁に出現する単語ほど、重要な単語であると考えられ、TFはその尺度となります。IDFとは、Inverse Document Frequencyです。これは、全文書数 (=N) のうち、単語iを含む文書数 (=DF) の割合を求め、その逆数をとったものです。通常は、更に、その対数をとってIDFとします。どの文書にでも出てくる単語は、文書を区別するには重要度が低いとして、IDFは、特定の文書にのみに存在する単語を重要な単語と考えます。

TF・IDFは、TFとIDFの積になります。TFが大きければ、その文書に出現する頻度が高いということで、IDFが大きければ、他の文書にはあまり出現しないということになります。このため、TF・IDFが大きい単語は、特定の文書にのみ、数多く出現する単語ということになり、重要度が高いと考えることができます。TF・IDFは、重要なキーワードの抽出や、検索の際の順位付けなどでも用いられます。

文書を検索する場合に、複数のキーワードを用いた類似度を定義して検索を行う方法もあります。この場合、検索キーとして文章を入力して検索するといったことが可能になります。このような複数のキーワードを用いた類似度を定義し、文書を検索する手法としてはベクトル空間法があります。

ベクトル空間法では、例えば、N個のキーワードについて、TF・IDFを並べたものを文書の特徴として利用します。また、検索キーワードも同様に、1つのベクトルとして表現します。

図3-3では、検索キーワードと、それぞれの文書のベクトルを矢印で表現しています。入力された検索キーワードと類似する文書を検索する場合、キーワード間の類似度を計算する必要があります。ベクトル空間法では、2つのベクトルの間の角度が小さいほど、類似度は大きいとします。この図では、文書1と検索キーワードとの間の角度と、文書2と検索キーワードとの間の角度を比べると、文書1との角度の方が小さいことから、この場合、文書1が検索キーワードに類似していると判断します。このように、検索キーワードとデータベース中の全文書との類似度を計算し、類似度の大きい文書から順に文書を表示することで、類似する文書の検索を実現することができます。

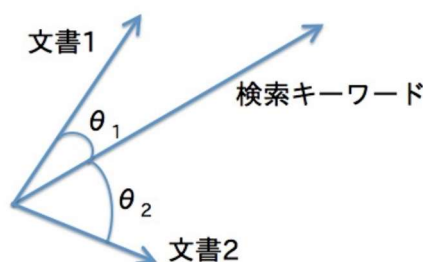


図3-3 ベクトル空間法

3.3 テキストマイニング

データマイニングは、多量に蓄積されたデータの中から知識を発見する処理です。このようなデータマイニングを、テキストデータに対して行うのがテキストマイニングです。テキストマイニングは、Webページを解析して評判を分析したり、自由記述アンケートを分析するといった場合などにも利用することができます。

例えば、自由記述アンケートの分析には、単語の共起性を利用することができます。共起性とは、共に起きる性質です。ある単語が出現した場合に、同時に別のある単語が出現することが多ければ、それらの単語の共起性は高いと言えます。例えば、筆記用具に関する自由記述のアンケートを行ったとします。そして、その結果を分析した結果、「図書館」という単語と、「音」という単語が同時に出現する場合が多かったとします。この場合、利用者が「図書館などの静かな場所で、筆記用具のカチツという音が気になっている」といった知識を抽出することができます。

また、類似する文書同士が近くなるように配置することで、内容によって文書を分類するといったことも可能になります。文書を類似度によって配置する場合、まず、対象となる文書から、キーワードの抽出を行います。そして、キーワードの出現頻度や、キーワードの重要度を計算し、それらをベクトルの形で表現したものを文書の特徴として利用することができます。このような多次元ベクトルとして表現された特徴量を使って、類似するもの同士が近くなるように配置するには、例えば、自己組織化マップや対応分析といった方法を利用することができます。これによって、Webページや、自由記述アンケートなどの文書データを、内容に基づいて分類することができます。

ここでは、例として、講義で使用されたスライド中のテキスト情報によって、スライドの分類を行ってみましょう。まず、対象となるスライドに含まれるテキスト情報を抽出し、形態素解析により単語ごとに分割します。そして、キーワードを抽出し、キーワードの出現頻度のベクトルを作成することができます。対象としたスライドからは、出現頻度が高い順に、「権利、情報、技術、倫理、論文」などのキーワードを抽出できました。

これらのキーワードの出現頻度のベクトルを用いて、スライドを2次元空間上に配置するには、対応分析を利用できます。対応分析により、類似するスライド同士を近くに配置できるとともに、関連するキーワードを重ねて表示することができます。

図3-4は、対応分析を用いて、テキストが類似するスライドを近くに配置した例です。例えば、画面の左中ほどに「倫理」のキーワード、画面左下には、「学術」、「論文」といったキーワードがあり、この部分には、主に学術論文の倫理に関するスライドが配置されていることが分かります。また、画面右側には、「著作」、「条」、「項」のキーワードが配置されており、この部分には、著作権の条項に関して述べているスライドが主に配置されていることが分かります。このように、対応分析を用いることで、内容の類似度に従って文書の分類を行うことができます。

このような、テキストマイニングを行うためのソフトにはさまざまなものがありますが、例えば、統計解析ソフトRから形態素解析ソフトMeCabを利用するRMeCabなどを利用することができます。

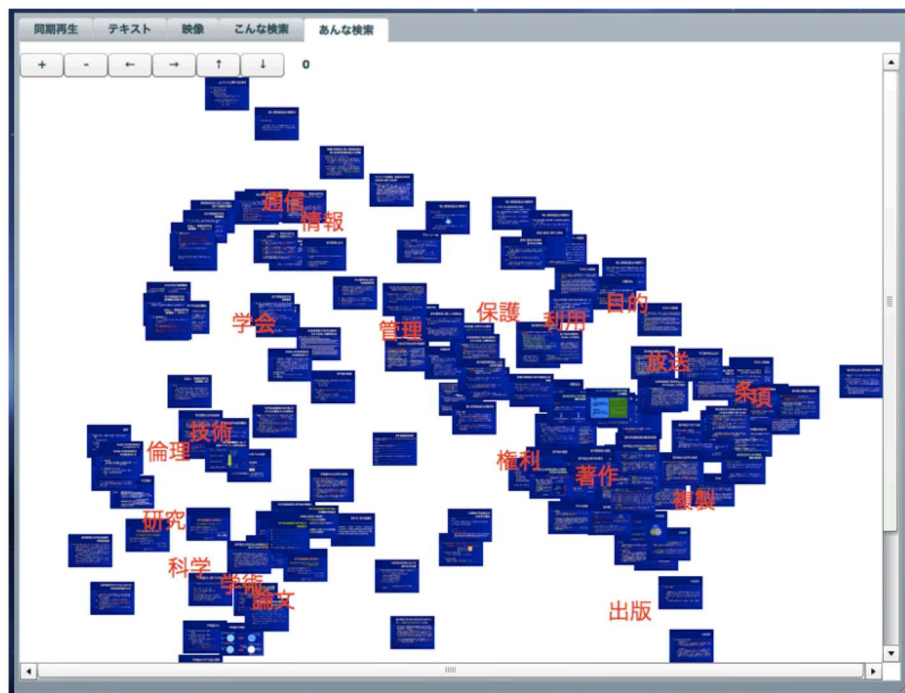


図3-4 対応分析によるスライドの分類