文脈を考慮した文書分類
―大規模データを深く解析するためのライブラリ群―

1. 背景

現在、Web 情報、ゲノム情報など、膨大な量の情報が整理できない状態で存在している。これらの情報を価値付けるため、情報を何らかの形で整理したり、また価値ある情報を抽出する必要がある。

既存の多くの自然言語処理や情報抽出は、単語単位で処理を行う。例えば、文書分類では文書中に含まれる単語の出現頻度でその文書を特徴付ける、いわゆる「bag of words」と呼ばれる手法が主流となっている。近年、多くの文書分類手法が提案されているが、いずれの手法も全て単語の出現頻度ベースで文書を分類していることには変わりはない。

この単語の出現頻度をベースにした自然言語処理には二つ問題点がある。

一つ目は、この処理は正確な単語分割が可能であることを前提としている点である。しかし、そもそも自然言語を最小な単語単位に分割した結果というのはその応用用途に大きく依存し、あらゆる場合に最適な分割は存在しないであろうことは近年指摘されている。

二つ目は、与えられた文書が単語のみから構成されてない場合は、この手法は適用できない点である。例えば、ゲノム情報上に単語の存在を仮定するのは難しく、提案されているゲノム情報のクラスタリング技術は単語分割を行わないものがほとんどである。

この二つの問題点を解決するため、文脈を考慮した自然言語処理を開発する。この”文脈を考慮する”という意味は、情報を単語に分解せずに処理する、既存の手法では不可能であった情報の並び方を網羅的に考慮することである。そして、文脈を考慮するために、あらゆる部分列を特徴として利用する方法を提案する。部分列を文書の特徴として利用する方法は近年盛んに研究されており、文書分類の精度向上や言語モデルの精度向上に大きく貢献していることが報告されている。この方法の実用上の問題点は、どのように任意の部分列を列挙し、情報を得るかである。例えば、転置ファイルを用いて任意の部分列を列挙、利用するのは計算量、領域量が指数関数的に増加するために、現実的ではない。また、任意の文字 2-gram からのクエリ拡張は機械学習への応用に耐えうる速度は達成できない。

Suffix Arrays, Suffix Trees は任意の部分列を高速に列挙可能なデータ構造である。しかし、これらを利用するには、必要領域量が問題となる。処理対象データが N Byte の時、Suffix Arrays は 5N ～9NByte、Suffix Trees は 10N Byte から 100N Byte 必要となり、対象とするデータが小さい場合のみしか適用できない。

本プロジェクトでは、数 GB から数 100GB の大規模な情報を処理するためのライブラリを提案、構築する。

2. 目的

本プロジェクトでは、Compressed Suffix Arrays(CSA)や Compressed Suffix Trees(CST)を基にしたライブラリ CS を開発し、それぞれの必要領域量を 0.5N Byte (Suffix Arrays)から 2N Byte(Suffix Trees)とし、通常の PC で数 GB から数 100GB のデータを処理することを目的と

する。このライブラリでは部分列頻度計数、文書頻度計数、周辺情報頻度計数、部分文字列復元などの操作を非常に高速に処理可能であり、単純な検索処理より高負荷な処理を行う機械学習やクラスタリングなどアプリケーションに耐えうるものとする。本プロジェクトでは従来の CSA や CST と比較し 10 倍高速な処理を可能とする新しい符号化を提案し実装する。

3. 開発の内容

(i) Compressed Suffix Arrays (CSA)

Suffix Arrays は検索対象文字列の全接尾辞を辞書式順序で整列させた時の各接尾辞番号を並べ保存した配列であり、Compressed Suffix Arrays はそれを圧縮させたものである。Suffix Arrays を用いることで任意の部分文字列検索、頻度計数等の処理を非常に高速に計算できる。検索対象文字列の長さが N Byte の時、Suffix Arrays は文字列長が N の時 5N～9NByte のメモリが必要なのに対し CSA ではその 1/10 程度の 0.3N～1N のメモリが必要である。

(ii) Compressed Suffix Trees (CST)

Suffix Trees は検索対象文字列の全接尾辞から構成された Trie を各節点が必ず 2 つの子を持つように枝を縮退させたものであり Suffix Arrays と比較し、さらに多くの文字列処理を高速に処理することが可能である。Compressed Suffix Trees はそれを圧縮したものである。検索対象文字列の長さが N Byte の時、Suffix Trees が 10N～100N Bytes のメモリが必要なのに対して CST ではその 1/5～1/50 程度の約 2N Byte のメモリが必要である。

(i)(ii)とも構築は、新たに開発した Incremental BWT + runLength と呼ばれる技術を利用している。これは、保存結果とほぼ同様程度の領域量でデータ構造を構築できる。図 1 にヒトゲノムに対して構築した時に必要としたメモリ使用量を示す。従来手法のように Suffix Arrays を経由して CSA,CST を構築する場合は 2.9GB*5 = 14.5GB の作業領域が必要であるが、本手法では 1.3GB 程度の作業領域で構築可能であった。

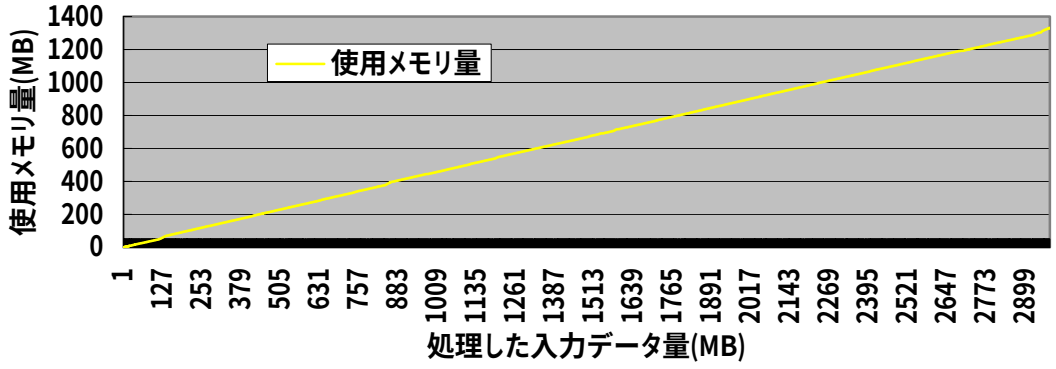

図 1　ヒトゲノム(2.97GB)に対するメモリ使用量

(iii) CSA/CST を用いたアプリケーション

CSA/CST を用いることで、情報を単語に様々な自然言語処理、機械学習を高速に行うことができる。その一つの例である文書分類では、文書の特徴として文書中に含まれる単語を用いるのではなく、任意の部分列を用いることで、分類に決定的となるフレーズや情報を取りこぼすことなく処理することが可能となる。また、固有表現抽出では、単語部分列の情報を扱うことが必要となるが CSA/CST を用いることで非常に大規模なデータに対しても高速に処理することが可能となる。その他、ゲノム解析においては非常に大規模なデータに対して部分列計数問題を高速に行うことが必要となるるが、CS を用いてそれらのアプリケーションを開発した。

その性能評価のデモとして、任意の部分列の検索結果ををリアルタイムで提供する検索サーバー/クライアントのソフトウェアを作成した（図 2）。これは Ajax を利用して作られており、殆どの検索結果を数 ms で以内で返す高速な操作を実現している。

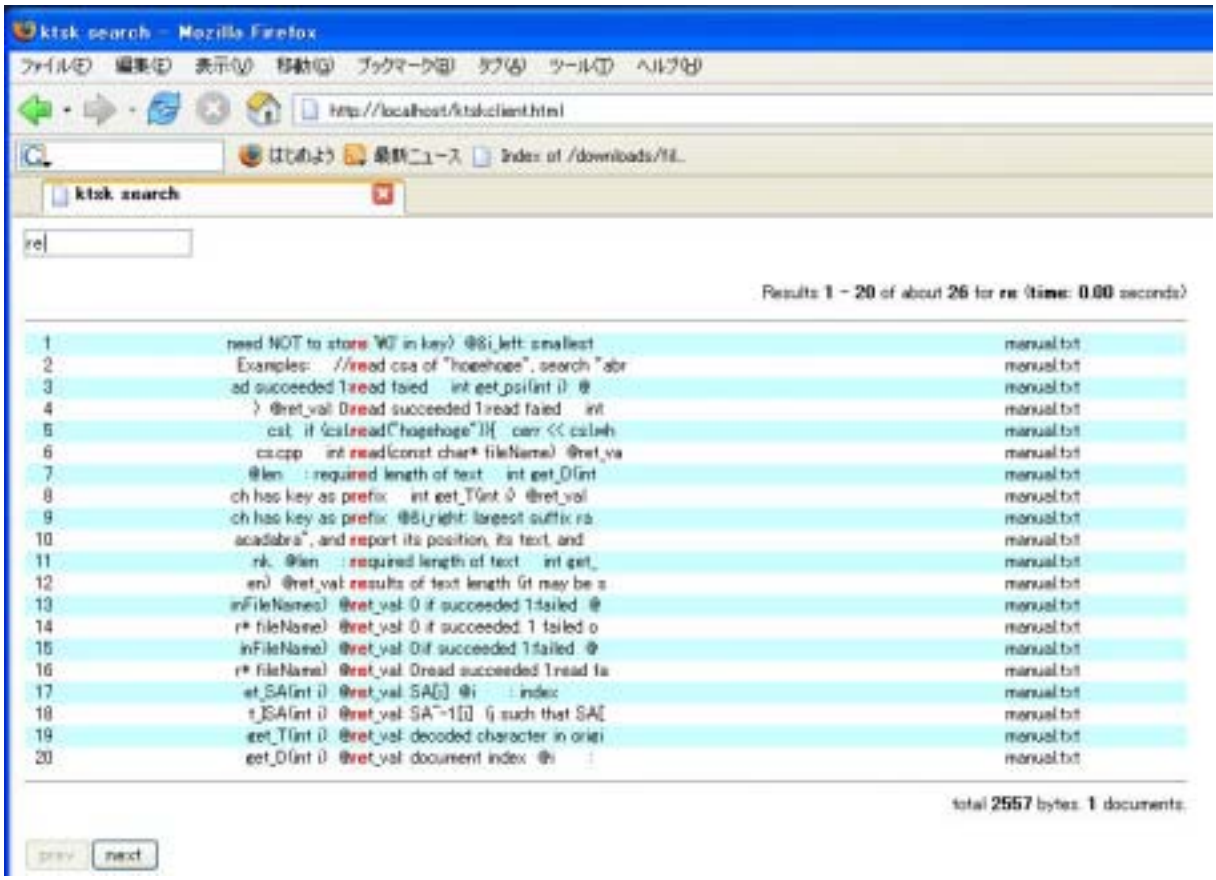

図 2　検索サーバー･クライアントウェア

4. 従来の技術（または機能）との相違

開発したライブラリは既存のライブラリの多くが Suffix Arrays や Suffix Trees であるのに対して、初めて現実的な圧縮付きの索引としての構築、利用を提供するライブラリとなる。このため、従来の技術と比較し 10 倍から 100 倍のデータを同一のマシン環境で実現可能となる。

また、圧縮を行うと、そのまま使った場合と比較し速度低下が発生するが、今回開発した Vertical Code 符号を利用することにより、従来提案されていた圧縮手法と比較し、約 10 倍

の高速化が実現できる。この他の理論上と、工学上のギャップを埋めるための技術も導入されて高速化、省メモリ化に貢献している。

5．期待される効果

開発したライブラリ群は高速な部分列計数等、各種の操作を実現するライブラリ群であり、一種のミドルウェアに位置する。そのため、直接アプリケーションとして利用されない。しかし、その応用範囲は多岐にわたり、結果として波及効果は非常に広いと考えられる。具体的なアプリケーションの利用例は次の通りである

・文脈を考慮した文書分類

計算量や領域量の制限を受けず任意の部分列を文書の特徴量として扱うことができるため、従来の文書分類と比較し、より高精度かつ応用範囲の広い文書分類の構築が可能であると考えられる。実際単語組み合わせの特徴量が評価文などの分類に非常に有効であることを国際学会等で示している。

・ゲノム解析

ゲノム配列のサイズは 1 生物種当たり数 GB、生物種間の比較を行おうとすると数十 GB とその規模は非常に大きい上に、従来自然言語処理が基にしている単語の存在が、ゲノム上では仮定できないため、ゲノム解析には多くのヒューリスティックスな方法が用いられていた。開発したライブラリを用いることで、近似やヒューリスティックスの適用を行わない、正確な結果が得られることが期待される。また、従来は数台のクラスタ群で解析を行うことが多いが、本提案手法では従来の PC(メモリ 4GB)でヒトゲノム全体が解析可能であり、実用的な面での効果も大きい

・統計的機械翻訳

近年目覚しい発展を遂げている統計的機械翻訳は、特に英中間、英アラビア間では実用的レベルに達し、その他の言語間でも開発が進められている。この中心技術となっているのが対訳コーパスにおけるフレーズ単位でのアライメントである。しかし、このアライメントで十分な精度を得るためには膨大な計算量を必要とするため、高速な部分列検索を行うデータ構造が必要とされてきた。本ライブラリを利用することで、十分な精度を高速で行うことが可能となる。

・情報抽出

膨大なウェブ情報や特許情報、技術論文集からの固有表現抽出や関係抽出が、近年盛んに研究されている。それらの手法では単純な単語単位の情報よりも粒度の細かい特徴量抽出が必要となり、そのため計算量、領域量ともに問題となっている。本手法を用いることにより部分列頻度計数や文書頻度計数など多くの操作を計算量、領域量の両面から効率良く行うことが可能となる。

6．普及（または活用）の見通し

ライブラリを公開する他、技術の詳細を論文で公開する予定である。一般的な利用者はプログラマー、特に基盤ソフトウェア関連、および、自然言語処理、機械学習、バイオインフォマティクス、などの技術者になると思われ、最終的な波及効果は大きいと考えられる。

7. 開発者名（所属）

岡野原　大輔　（東京大学情報理工学系コンピュータ科学専攻）
（参考）開発者ＵＲＬ
(プロジェクト専用 web ページは公開準備中です。
http://homepage3.nifty.com/DO/index.html
http://www-tsujii.is.s.u-tokyo.ac.jp/~hillbig/atlab-j.html