〈文書比較について〉

【はじめに】

本書は、日本語の文書比較機能のプログラムについて、記載するものである。今回公開している文書比較のプログラムは、非常に冗長的で、煩雑なプログラムとなっている。ですので、プログラムから、詳細な仕様を把握することが、非常に困難と考えられる、このドキュメントで、処理内容(文書比較のアルゴリズム)について、簡単に記述することで、ソースコードのメンテナンスに役立てて頂ければと思う次第である。

日本語文書の一致判定で難しいところは、文言の、文書内での前後への移動や、同じ文言を多数記述した場合に、一致位置を、誤って検出してしまうという点ではないでしょうか。これを如何に改善するかというところがポイントと思われ、私は、なるべく多くの連続した単語で、一致確定をするように作成しました。その、あたりを絡めながら、今回の、文書比較のアルゴリズムを関数ベースで、非常に簡単ではありますが、説明したいと思う。

【文書比較アルゴリズム】

文書比較を行うための、I/F となる関数は、**DfDiffSentence 関数**である。まずは、この関数についての、概要を簡単に説明する。尚、一部の関数名については、省略した記述となっている。

・DfDiffSentence 関数

この関数は、引数に渡された文書 A と文書 B の文字列を比較し、出力として DIFF 結果情報を StructDiffInfo 構造体に格納するものである。処理内容を簡単に、時系列に、記述すると以下となる。

1. 文書 A と文書 B を解析し、それぞれ単語分割する。また、それぞれの文書を、句点やクエスチョンマークなど文末特有の記号を一区切りとした、文字列に分割管理する。これを本書では、文と表現する。
2. 文書 A と文書 B が、同じ文書構造(文の数と相関関係により)かどうかを判定し、同じと判定された場合は、文書 A と文書 B における、一文ずつを比較する(**DfDiffSentence2 関数**)。そうでない場合は、**DfDiffSentence1 関数**により比較を行う。

上記の 2 の手順で、DfDiffSentence1 関数と DfDiffSentence2 関数の 2 つに処理分岐しているが、それぞれぞれについて、次に説明する。

・DfDiffSentence1 関数

この関数が基本 DIFF 処理で、文書 A と文書 B 全体を比較する処理となっている。以下が、DfDiffSentence1 関数の処理概要である。

1. 呼び出し元の処理で文単位に分割したデータを A、B それぞれで、一致検出する。一致が確認できた場合は、該当文の該当単語群を一致と確定する。この処理が、Step0 関数により行われている。
2. 次に、_Step1 関数で、あるルールに基づき、連続した単語(文言)で、文書 A と B の一致箇所を検出する。このタイミングの検出では、ルールには、沿っていないが、文書 A と B で、他にも、一致する文言が存在する可能性が残っている。これらについては、後の、_Step2 関数により、さらに、一致性をチェックする。
3. 上記までの処理で、未確定(不一致)の単語が連続して続いている個数をチェックし、その最大値を算出する。この最大値は、次の_Step2 関数で使用される。
4. _Step2 関数により、未確定の単語群の一致をさらに、検出する。ここまでの処理で、DIFF 結果出力の為に必要な、文書 A と文書 B の一致箇所がすべて抽出されたものとする。
5. 最後に、ここまでの処理で、検出された情報を元に、DIFF 結果情報を出力する。_Step3 関数。

・DfDiffSentence2 関数

この関数は、DfDiffSentence1 関数とは異なり、文書 A と文書 B における 1 文ずつを比較し、それぞれの結果を結合することにより、一つの文書 A、B の DIFF 結果を検出するものである。ですので、お互いに、対応する文が、それぞれ存在している必要がある。

文書 A:私は、昨日、朝 6 時に起床した。　… ①
その後、パソコンで、仕事をした。　… ②

文書 B:私は、明日、朝 6 時に起床する予定です。　… ①
その後、パソコンで、仕事をするつもりです。　… ②

このような文書があった場合に同じ文書構造と判定し、①同士、②同士を比較し、その結果を結合して文書 A と文書 B の DIFF 結果とする仕様である。

続いて、DfDiffSentence1 関数内で、呼び出している、それぞれ関数について説明する。また、DfDiffSentence2 関数の処理は、関数内で、1 文ごとに DfDiffSentence1 関数を呼び出すことによる、比較であるため、DfDiffSentence2 関数の詳細については、割愛する。

・DfInternalDiffSentence_Step0 関数

文書 A と B の文分割された、それぞれのデータで、一致検出を実行し、一致確認できたものを一致と確定する。簡単に言うと、文単位での一致を検出する処理である。

・DfInternalDiffSentence_Step1 関数

この関数は、DIFF 処理の第二段階として、文書 A と文書 B の一致箇所を抽出する関数となっている。

まず、初めに、変数 x のループが存在するが、このループは、文書 A の単語群によるループである。(単語(x)は、文書 A の x 番目の単語を表すものとする。)

このループでは基本的に、文書 A の単語(x)と、文書 B 中の単語と一致するものがあるかどうかをチェックするものであるが、単純に一致だけをチェックすると以下のような文書の場合に問題が発生する。

文書 A: **湯気**が、立ち上っている。湯気が釜から立ち上っている。湯気が、鍋から立ち上っている。

文書 B: 立ち上っている。**湯気**が釜から立ち上っている。湯気が、鍋から立ち上っている。

文書 A の文頭の″湯気”が、文書 B 中の”釜”の直前に存在する**″湯気”**に誤認一致してしまう可能性がある。ですので、まず、先頭の″湯気”と一致するものを**文書 A 中の単語からリストアップ**する。

リストアップの際に、″湯気”に続く、単語も同時にチェックし、単語(x)との比較により、お互いが、同じでなくなるまでを、一つの区切りとした文言を抽出する。このリストアップされた、文言の**最も大きい単語数**、で、文書 A の単語(x)からの連続した文言と、文書 B における文言のチェックを行い、一致すれば、一致確定とする。

※但しこの処理は、単語(x)が、句読点と平仮名 1 文字の場合は、この処理から除外してある。

そうすると、上記の例文では、以下の 3 つがリストアップされる。ここでは、分かりやすくするため文字列で表記しているが、プログラム内では、**一致個数**で管理している。(比較するときは、一致個数+1 の文言で、一致比較しているので、以下のような表記となっている。)

″湯気が、”、″湯気が釜”、″湯気が、鍋”

この中で、単語数が最も大きいのは、4 である。ですので、4 個の連続した単語を一くくりとした文言で、文書 A と文書 B での一致性をチェックする仕様としている。

ここまでの処理を、文書 A の単語(x)すべてに対してループ処理で行っている。

**** 仕様変更 v. 103 での変更 ****

v. 102 まででは、リストアップした全てに対してチェック処理を行っていたが、設計ミスの為、処理を削除しました。

例えば、リストアップされたものが、以下であれば、最後の″湯気の”と″湯気の”は、単語(x)との比較で、抽出されたものであり、″湯気の”と″湯気の”は同じ物なので、これらを一致比較してしまうと、誤認一致する可能性がある。

″湯気が、”、″湯気が釜”、″湯気が、鍋”、″湯気の”、″湯気の”

****

__Step0 と_Step1 関数により、お互いの文書で、一致したものと、そうでないものものが(歯抜けの状態)となっている。お互いの文書における、未確定(一致ではない)単語の中には、まだ、一致する可能性が残っている単語も存在する。それを_Step2 関数においてチェックする。

・DfInternalDiffSentence_Step2 関数

この関数は、_Step0 関数、_Step1 関数で、未確定となっている単語をさらにチェックし、一致確定できるものがあれば、確定していく処理となっている。

本関数が、呼び出される前に、文書 B 側で連続して未確定となっている単語の数の最大値(max_cnt)を算出しており、この最大値数、単語が連続して一致するものがお互いに、ないかチェックする。一致すれば一致確定とする。

このことを、max_cnt を 1 ずつ、デクリメントし、max_cnt=1 となるまで、繰り返し処理する。

```
Loop Start max_cnt > 0
        ** 一致チェック処理。 **
        ** 記号、制御コードと平仮名 1 文字は、このチェック処理で確定しない場合がある **

        max_cnt = max_cnt - 1
Loop End
```

連続数の大きいものを優先的にチェックを行うことで、複数単語での一致となり、句読点や平仮名 1 文字を誤認一致を防ぐことができる。

ここまでで、句読点の 1 文字が確定されていない可能性があるので、最後にもう一回だけ、句読点の 1 文字を後方一致により確定処理を行っている(一つ前の一致位置より、文書 B 側の後ろ側だけチェック)。

最後まで、未確定で残った平仮名の１文字は、未確定として放置し、DIFF として判定する方が、望ましいと思われる。データとしては、一致するものがある場合もあるのですが、位置誤認の可能性がありえる。

文書A: AとBは、Cである。Aが、Dに入れ替わった。

文書B: A、Bは、Cである。Aが、Dと入れ替わった。　　→ 誤認の可能性。

・DfInternalDiffSentence_Step3 関数

_Step2 関数までの処理で、文書A と文書B の一致箇所がすべて抽出された状態となる。本関数では、この抽出された情報を元に、DIFF 結果情報を作成するものである。処理内容を大きく分類すると、以下の２つとなる。

- 【削除】、【挿入】、【変更】の DIFF 結果判定。
- 文言の【移動】を判定できるよう、文言のグループ化(インデックスを割当てる)。

文頭と文末の DIFF 結果判定は、特殊な処理となるので、関数の先頭にて行っている。ここでは、以下の３つの可能性をチェックし、【変更】と判定できるものを確定している。但し、このタイミングでは、【仮変更】として、確定し、後の処理で【変更】確定とする仕様である。

- 文頭同士の【変更】
- 文末同士の【変更】
- 文書A と、それに対応した文書B の途中の【変更】

文頭と文末のチェック処理が終了したら、基本的な DIFF 結果判定を行う。これが、終了しても、文書A 側に、未確定(Unknown、idx_b が負)で、DIFF 判定されていない情報が残っている可能性がある。これらは、すべて、文書A 側から【削除】されたと判定できる。

この【削除】情報を、文書B 側に追加する。この際、ソースとしての文書B 情報(snt_b)は残しておきたいので、文書B の情報のコピー(tmp_inf)を作成し、それに追加するようにしてある。

tmp_inf に、DIFF 結果を作成するための、必要な情報がすべて揃ったことになり、最後に、この情報を元に、DIFF 結果情報を作成する。この際、【移動】判定するために、必要なグループ・インデックスも一緒に、付与している。

【相関係数について】

DIFF 比較処理の処理分岐に、相関係数の値を利用しているが、これについて、少しだけ、補足くしておく。

文書A の各単語を、文書B 内から検索するものとし、文書A の各単語の位置を x、見つかった文書B での位置を y とする。ちなみに、見つからなかった単語は、相関係数の算出からは、除外している。

文書A の各単語の位置インデックス(x)が大きくなれば、文書B で見つかった位置インデックス(y)も大きくなるという関係があれば、統計学的に、x と y に正の相関があるといえる。

もし、文書A とB に含まれる文の数が同じで、上記の、正の相関があれば、文書A とB は、少なからず、DIFF が存在する可能性があるが、おおむね同じ文書といえる。言い換えると、お互いを構成している各文が、前から順番に対応していて、文自体の、移動(入替)は存在していないと、言えるのでは、ないでしょうか。この場合、１文ずつを比較する **DfDiffSentence2 関数**を呼び出し、逆に、相関関係がない場合は、**DfDiffSentence1 関数**を呼び出すことにより、DIFF の抽出を実現している。

今回のプログラムは、文書に応じて、異なる DIFF 抽出方法を対応させていることになり、このことに、少し違和感を感じる方がいらっしゃるかもしれません。私は、DIFF の結果が、ユーザに分かりや

すい方が良しと、考え、この点を重視し、文書の相関係数値によって、DIFF 結果の抽出･判定方法を変えるという仕様を、あえて採用しました。

文書比較の結果を、どのようなシステムに、どのように利用･応用するかということを考慮し、このあたりの仕様を、皆さんの方で、柔軟に変更して頂ければ、と思っている。

【用語定義】

文書比較機能を実装するにあたり生成される、ドキュメントや、プログラムのコメントなどに記載される用語で、一部、文書比較用に、用語の意味を独自に定義しているものがある。それら用語を以下に纏める。

文字：文字列を構成する単一、すなわち文字列長が 1 の文字列。

単語：同一種の連続した文字を、ひとくくりにした文字列。

文言：複数の単語を連結した文字列のこと。

文節：句読点をひとつの区切り(終端)とした文言のこと。

文節接尾辞：日本語文章において、文節の末尾に記述されることの多い。記号のことで、
″、。，．・？！）］｝」』!),.:;?]｝。」、″ などがある。

文：複数の文言から文節から構成され、句点やクエスチョンマークなどの文末特有の
接尾辞(改行を含む)で、終端する文言。
″。。..！!??)）]］｝｝」」』″ & vbLf

文書：複数の文節や文から構成された文字列のこと。

文頭：文書の先頭。

文末：文書の末尾。

以上