技術文書推敲支援システムにおけるわかりにくい文の検出機能の改善

Improvement of a detection function of confusing sentences in the technical writing support system

テーマ：インターネット技術とその応用
指導教員：松本章代

教養学部　情報科学科
1257253 山崎 隼永

1. 研究背景および目的

技術的な文書を作成する際，論理的かつわかりやすい文を書く能力が求められる．学生にとっても，レポートを書くにあたってこの能力が必要不可欠であるのだが，学生の文書作成能力は近年低下傾向にあるとされている．このような能力を身につけ，学生のレポートの質を向上させるためには，指導者に文書を添削してもらうことが有効である．しかし，指導者一人に対して学生の数が多いことが大半であるため，学生が十分な指導を受けることができないのが現状である．

そこで本研究室では，数年前から技術文書推敲支援システムというウェブサービスの開発を行っており，初年次教育などの授業で実際に利用させている [1]．レポートを書くのに慣れていない学生に対して，提出する前に自分の文書を推敲・校正することを支援するものであり，学生の文書作成能力を向上させることを目的とした教育システムである．本システムには，主に 3 つの機能がある．(1) 文書の基本ルールに沿って書かれているかのチェック機能，(2) 一義的で簡潔性の高い文章に修正するための指摘機能，(3) 論理性を高めるための推敲を支援する全体の流れの可視化機能である．

このシステムの (2) の機能である文章の指摘機能には，わかりにくいとされる文章を検出する機能がある．しかし，そのわかりにくいと判定するルールがヒューリスティックに決められている．そこで本研究では，わかりにくい文のルールに客観的な判断を持たせることを目的とし，機能の改善を行う．

2. 関連研究

文書の推敲を支援するシステムはすでに存在している．菅沼ら [2] らは「マニュアルなどの文章は内容が読み手に一意に伝わる文を書くべきである」と考え，書き手の意図が読み手に正しく伝わらないような文の抽出を行っている．本システムでは機械学習により誤解されうる文を判断するのに対し，菅沼らは可能な係り受けをすべて見つけ，不自然な解釈になる係り受けを取り除くことで，抽出精度を向上させている．

3. システム概要

技術文書推敲支援システムは，利便性を考慮してウェブアプリケーションとして構成されている．システムは Linux 上で動作し，開発言語には Ruby を用いている．また，日本語の係り受け解析には CaboCha を用いている．本研究で作成したシステムは，最終的にこのウェブアプリに組み込まれることになるため，同様の開発環境で研究を行う必要がある．

4. わかりにくい文の判断

木下 [3] は，わかりやすく簡潔な表現にするための注意点として，次の 5 つを挙げている．(1) 文を頭から順に読み下してすらすらと文意が通じるように書けていれば，長さにこだわらなくていい，(2) 格の正しい文を書く，(3) まぎれのない文，誤解されないような文を書く，(4) 簡潔に書く，(5) 文の区切り記号を正しく使う，である．また阿部 [4] は，簡潔な文を書くために，1 つの文に 2 つ以上のことを詰め込まない，逆説の意味を持つ「が」以外は，2 つ以上のことがらを「が」で続けて 1 つの文にしてはいけない，などを挙げている．しかし，これらのルールは人が文章を書くときに注意することを想定しているものであるため，文の意味に踏み込む必要があるものばかりである．プログラムで再現することは，不可能であると判断した．そこで，著者はまず長くてわかりにくい文を収集し，そこからルールを見出すという逆側からのアプローチをかけていく．

4.1 学生レポートの解析

表 1.　研究で使用した文書一覧

記号	年度	総文数	80 字以上
s1	2015 年	1330 文	128 文
s2	2014 年	1688 文	71 文
s3	2013 年	1647 文	103 文

表 1 は本研究で使用した文書一覧である．(s1)-(s3) はどれも初年次教育の授業における学生レポートである．学生が書いたレポートに対し，人手による係り受け解析と，プログラムによる係り受け解析の実行結果を比較する．係り受け解析とは，日本語の文における主語–述語の関係や，修飾–被修飾の関係を対応させることである．本研究ではまず (s1) から 80 字以上の文だけを抽出し，解析を行った．(s1) から抽出した 128 文を CaboCha に渡し，日本語の係り受け解析をさせ，著者が自ら解析を行い，結果を比較したところ，CaboCha の係り受け解析は完璧ではないことが分かった．間違った解析を行った文を見ると，人間が読んでわかりにくいと感じる複雑な文であることがわかった．この解析の結果から，著者は「CaboCha が解析を間違える文がわかりにくい文である」という仮説を立てた．

4.2 決定木の生成

表 2 は (s1)-(s3) の解析結果を，CaboCha が正しく解析できた文を True，誤った解析をした文を False で表した表である．CaboCha が解析を間違える文の条件はどのようなものなのかを判断する．例えば文節の数や修飾節の数などであれば，何個以上あると解析が上

表 2.　学生レポートの解析結果

記号	80 字以上	True	False
s1	128 文	116 文	14 文
s2	71 文	59 文	10 文
s3	103 文	91 文	12 文

```
yousetsu <= 0 : True (19.0)
yousetsu > 0 :
|   setsuzokujoshi > 0 : True (147.0/16.0)
|   setsuzokujoshi <= 0 :
|   |   yousetsu <= 2 :
|   |   |   yousetsu <= 1 : True (16.0/3.0)
|   |   |   yousetsu > 1 :
|   |   |   |   doshi <= 5 : True (5.0)
|   |   |   |   doshi > 5 :
|   |   |   |   |   doshi > 7 : True (3.0)
|   |   |   |   |   doshi <= 7 :
|   |   |   |   |   |   bunsetsu <= 19 : True (2.0)
|   |   |   |   |   |   bunsetsu > 19 : False (2.0)
|   |   yousetsu > 2 :
|   |   |   taisetsu <= 1 : True (2.0)
|   |   |   taisetsu > 1 : False (3.0)
```

図 1.　決定木

手くいかないのか，そのような if 文の条件式を判断するために，c4.5 のアルゴリズム [5] を用いて決定木の生成を行った．決定木を生成するためには，条件分岐させるための判断材料が必要である．決定木の判断材料として，単語の数，文節の数，連用修飾節の数，連体修飾節の数，動詞の数，接続助詞の数を用いることとした．(s1) の 128 文に対して 1 文ずつ上記の 6 項目を数えさせ，その文が True なのか False なのかを書きだした．c4.5 のアルゴリズムを用いたプログラムにより，その文章が True であるのか False であるのかを判別する決定木の生成を行った．しかし，生成された決定木は『True』であった．そうなってしまった原因は，データに問題があった．表 2 の (s1) を見ると，128 文に対し，False の文が 14 文しかなく，8 割以上が True であったことが原因と考えられる．そのため，急遽 (s2) のデータの解析を行い，True と False の仕分けを行った．(s2) のデータをプラスし，改めて実行したところ，条件分岐されている木を生成することができた．

4.3　判別プログラムの作成

図 1 は c4.5 のアルゴリズムを用いて作成したプログラムの実行結果である．変数の意味は，tango(単語の数)，bunsetsu(文節の数)，yousetsu(連用修飾節の数)，taisetsu(連体修飾節の数)，doshi(動詞の数)，setsuzokujoshi(接続助詞の数)，である．生成された決定木を Ruby の if 文の形式に書き直し，判別プログラムを作成した．プログラムを作成するときに用いた (s1) と (s2) のデータの合計 199 文に対して実行すると，表 3 の結果になった．これがこの生成された決定木の実力である．

5.　評価実験

作成した判別プログラムに対して，評価実験を行う．プログラムを作るのに使用したデータを用いる closed テストではなく，未使用のデータを用いる open テストを行う．今回はまだ使用していない (s3) のデータに対し，人手による判別結果と判別プログラムによる実行結果を比較する．結果を表 4 に示す．

表 3.　closed テスト

		システム	
		True	False
目	True	175	0
視	False	19	5

表 4.　open テスト

		システム	
		True	False
目	True	87	4
視	False	12	0

今回このような結果になってしまった原因として考えられるのは，データの数が少なすぎたことがあげられるであろう．open テストと closed テストで用いたデータはどれも初年次教育で提出されたレポートであるため，差があるとは考えられない．また，生成した決定木は Error が 19 個もあるので，決定木が妥当のものではないことも挙げられる．決定木を生成する際の判断材料に単語の数を用意したが，実際に生成された決定木には使われていなかった．わかりにくい文を検出するのに単語の数を考慮する必要がないことが分かった．判断材料の再検討が必要である．

6.　まとめ

「CaboCha が解析を間違える文はわかりにくい文である」と仮説を立て，研究を進めてきたが，今回の実験の結果ではそれを判断することができない．このテーマをやり直すとしたら，データを 80 字以上の文に限定してしまっている点を改善し，より多くのデータで決定木を生成した場合，どのような結果になるのかを調べる必要がある．また，c4.5 以外の判別手法（機械学習）を用いた場合の結果も検討する必要がある．

参考文献

[1] 松本章代：科学的文書の推敲・校正を支援する教育システムの構築，東北学院大学教養学部論文集 No.167，pp.53-62 (2014)．

[2] 菅沼明，小野貴博：文書推敲支援における読み手に誤解される文の抽出，情処研報 2007-DD-61，Vol.2007，No.50，pp.31-38(2007)．

[3] 木下是雄：理科系の作文技術，中公新書 624，pp.118-152 (1981)．

[4] 阿部圭一：明文術-伝わる日本語の書き方， NTT 出版，pp.122-144 （2006）．

[5] J. R. Quinlan. Improved use of continuous attributes in c4.5. Journal of Artificial Intelligence Research， 4:77-90， (1996)．