Support Vector Machineを用いたコーパスの誤り検出

中川 哲治　　松本 裕治
奈良先端科学技術大学院大学 情報科学研究科
{tetsu-na,matsu}@is.aist-nara.ac.jp

1 はじめに

近年，自然言語処理においてコーパスを利用した研究が盛んに行なわれている．コーパスに基づくシステムの性能は，利用するコーパスの量と質に大きく左右されるため，できるだけ良質なコーパスが使えることが望ましい．しかしながら，コーパスは人手で作成されるため多少の誤りを含んでおり，これらはコーパスを利用した研究を行なう上で問題となる．そのため，コーパス中の誤りを見つけ出して修正する必要があるが，通常コーパスの量は膨大であるため，人手でその作業を行なうのは困難である．そこで，コーパス中の誤りを自動的に検出する技術が必要とされる．

コーパスの誤り検出というタスクを考えた場合，そのための訓練データは通常存在しないため，教師なし学習問題として解決しなければならない．コーパスは一般に何らかの基準に基づいて作成されるため，一貫性を持っていると考えられる．もしその一貫性を乱すような例外的な要素が見つかれば，それは誤りである可能性が高い．そこで，本稿ではコーパスの誤り検出を，コーパス中で一貫性を乱すような例外的な要素の検出として考える．そのような例外的な要素の検出に，機械学習アルゴリズムの一つである Support Vector Machine (SVM) を使うことができる．SVMは，学習事例の一つ一つに対して重みを与えるが，特に例外的な事例や重要な事例に対しては大きな重みを与える．そこでこの特徴を利用し，例外的な事例を検出することでコーパスの誤り検出を行なう方法を次の節で述べる．さらにそれを発展させ，データ中から非一貫性を抽出することで，実用的なコーパスの誤り検出を行なう手法を3節で検討する．複数のコーパスに対して実験を行ない，本手法で形態素情報の誤り検出を高い精度で行なえることを4節で示す．

2 Support Vector Machineを用いたコーパスの誤り検出

この節では，SVM[3] を利用してコーパス中から例外的な事例を検出する方法を述べる．

SVMは，素性ベクトル $\mathbf{x}_i \in \mathbf{R}^L$ とラベル $y_i \in \{+1,-1\}$ の組からなる l 個の訓練事例に対して，次の式を最適化するような α_i を求める:

$$\text{minimize} \quad \frac{1}{2}\sum_{i,j=1}^{l}\alpha_i\alpha_j y_i y_j K(\mathbf{x}_i,\mathbf{x}_j) - \sum_{i=1}^{l}\alpha_i,$$

$$\text{subject to} \quad 0 \le \alpha_i \le C,$$

$$\sum_{i=1}^{l}\alpha_i y_i = 0.$$

ここで，関数 $K(\mathbf{x}_i,\mathbf{x}_j)$ は素性ベクトル $\mathbf{x}_i,\mathbf{x}_j$ を適当な非線形関数 $\mathbf{\Phi}(\mathbf{x})$ によって高次元空間へ写像し，そこでの内積を返す関数でカーネル関数と呼ばれるものである $(K(\mathbf{x}_i,\mathbf{x}_j) = \mathbf{\Phi}(\mathbf{x}_i)\cdot\mathbf{\Phi}(\mathbf{x}_j))$．定数 C は訓練誤差を調整するパラメータで α の値の上限を定める．この二次計画問題は，カーネルを使って写像される高次元空間上で，訓練データを線形分離する平面でマージン (分離平面と事例ベクトルの距離) が最大になるようなものを求めている．テスト事例 $\mathbf{x}$ に対するラベルは，テスト事例と訓練事例の内積に α_i で重みをつけて足し合わせた値の符号で求められる:

$$\text{sgn}\Big(\sum_{i=1}^{l}\alpha_i y_i K(\mathbf{x}_i,\mathbf{x}) + b\Big).$$

ここで b はしきい値である．このように，SVMは学習時に各訓練事例に対して α_i という値を計算するが，この値は重要な事例や例外的な事例に対して大きな値を持つ．そこで，この値を利用することでコーパス中の誤り (例外) を検出できる可能性がある．

コーパス中の形態素情報の誤り検出をSVMを用いて行なうには，コーパス中の各形態素を一つの学習事例として，形態素解析を行なうためのSVMのモデルを学習させ，α に大きな値の付与された形態素を取り出せばよい．ここでは，SVMにより形態素解析を行なう方法として，修正学習法 [8] を用いた．これは，訓練データ中の形態素を正例の事例，確率的モデル (品詞n-gramモデル等) が解析に失敗した形態素を負例の事例として学習データを作成し，確率的モデルの誤りを修正するようなSVMのモデルを作成する方法である．例として，「急/名詞 | に/助詞-副詞化 | 帰る/動詞」という文に対して，確率的モデルが「に」を「助詞-格助詞」と誤って解析するような場合，以下のような訓練事例が生成される．

<クラス (ラベル)>	<素性ベクトル>
名詞 (○)	(word:急, word-1:BOS, ...)
助詞-格助詞 (×)	(word:に, word-1:急, ...)
助詞-副詞化 (○)	(word:に, word-1:急, ...)
動詞 (○)	(word:帰る word-1:に, ...)

このように各クラス (品詞) に対して，正例と負例からなる訓練事例が作成される．

SVM で使用する素性として，日本語の形態素解析では次のものを使用した．

1. 注目している位置の形態素の単語と活用形
2. 前 2 つの形態素の単語と品詞と活用形
3. 後 2 つの形態素の単語と品詞

英語の品詞タグ付けも日本語の形態素解析と同様に行なうことができるが，素性としては次のものを使用した．

1. 注目している位置の単語とその 4 文字までの語頭と語尾，数字・大文字・ハイフンの有無
2. 前 2 つの単語と品詞
3. 後 2 つの単語と品詞

3 コーパスからの非一貫性の抽出

前の節では，SVM を用いてコーパス中の例外的な要素を検出する方法を考えた．しかし，この方法を直接コーパスの誤り検出に適用するにはいくつかの問題がある．一つは，例外的な事例が必ずしもコーパスの誤りというわけではなく，例外的な正しい事例も数多く存在することである．もう一つは，たとえコーパスの誤りと思われる箇所のみを提示しても，それが本当の誤りかどうかを人手により判断するのは難しい場合が多いことである．そこで，この節ではそれらを解決する方法を考える．

上述の問題点を解決するには，コーパス中で例外的な事例を一つだけ提示するのではなく，その例外的な事例と対になってコーパス中の一貫性を乱すような別の事例も同時に提示すれば良い．そうすれば，どちらか一方が誤りである可能性は高く，また人間も判断を行ないやすい．もし似た素性を持った 2 つの事例があり，それらが異なったラベルを持っていた場合は，一貫性を乱していると考えられる．ここで，SVM における 2 つの事例 $\mathbf{x}_i, \mathbf{x}_j$ の類似度は，次のようにして計算できる:

$$d(\mathbf{x}_i,\mathbf{x}_j)=\sqrt{\|\Phi(\mathbf{x}_i)-\Phi(\mathbf{x}_j)\|^2},$$
$$=\sqrt{K(\mathbf{x}_i,\mathbf{x}_i)+K(\mathbf{x}_j,\mathbf{x}_j)-2K(\mathbf{x}_i,\mathbf{x}_j)}.$$

結局，前節の方法により例外的な事例 $\mathbf{x}$ が検出された場合，$\mathbf{x}$ とは異なるラベルを持ち，かつ $d(\mathbf{x},\mathbf{z})$ があるしきい値 θ_d 以下であるような事例 $\mathbf{z}$ を抽出して $\mathbf{x}$ と $\{\mathbf{z}\}$ の対を提示することで，コーパスの誤り検出を行なうことができる．

4 実験

RWCP コーパス，京大コーパス，Penn Treebank WSJ コーパスの 3 つのコーパスを使って，本手法を用いたコーパスの誤り検出の実験を行なった．以下の実験では，SVM の kernel 関数には 2 次の polynomial kernel を使用し，α_i の上限値 C には 1.0 を用いた．

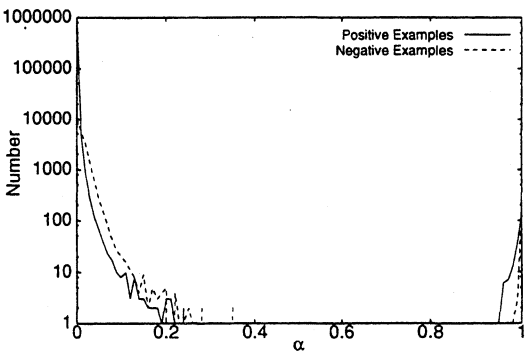

図 1: RWCP コーパスでの α_i の値の分布

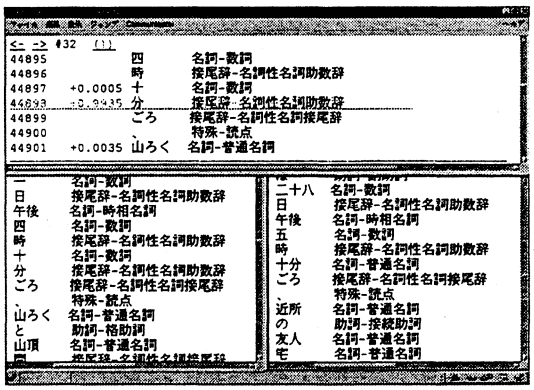

図 2: コーパスの誤り検出用ツール

4.1 RWCP コーパスでの誤り検出

コーパスとして，37,589 文 (934,034 形態素) を含む RWCP コーパスを使用した．まず，修正学習法を用いて形態素解析を行なうための SVM のモデルを学習させた．この時の α_i の値の分布は，図 1 のようになった．α_i の値は，0 と C の周辺に集中して分布しており，C に近い α_i の値を持つ事例は例外的な事例と思われる．そこで，α_i の値が 0.5 以上の事例を例外的な事例とみなした．また，θ_d の値は 0 とした場合でも，十分にコーパス中の非一貫性を抽出できたため，$\theta_d = 0$ とした．なお，以下の実験では全てこのパラメータを使用した．このような条件でコーパスの誤り検出を行なった結果，177 個の形態素が誤りとして検出された．

コーパス誤り検出のためのツールを HTML を使って作成した (図 2)．ウィンドウの左下部に例外的と判断された形態素付近のコンテキストが，右下部にそれと共に一貫性を乱すと判定された形態素付近のコンテキストが表示されている．

誤りとして検出された形態素に対して，それが本当に

表 1: RWCP から検出された誤りの内訳

	数	割合
誤って検出された数	11	6%
正しく検出された数	166	94%
(品詞付与の誤り)	(120)	(68%)
(形態素分割の誤り)	(46)	(26%)
合計	177	100%

コーパスの誤りなのか人手で確認を行なった．その結果が表 1 である．ここでコーパスの誤りは，形態素区切りの誤りと品詞付与の誤りの 2 つに分けて調べた．結果，システムの出力に対する正解の割合 (精度; precision) は 94%となった．また，品詞付与の誤りの方が，形態素分割の誤りよりも多く検出された．

正しく検出された例と誤って検出された例を表 2 に示す．誤りとして検出された形態素は下線で示してある．正しく検出された例からは，形態素分割と品詞付与の両方の誤りが検出できていることが分かる．一方，誤って検出された例は，本手法の問題点を示している．この実験では，注目している形態素の前後 2 つの形態素を SVM の素性として使用した．表 2 の誤って検出された例では，前後二つの形態素は同じであるにもかかわらず，注目している位置の形態素の品詞だけが異なっている．このような場合に対して，SVM は 2 つの形態素の違いを区別をすることができず，一貫性がないとみなしコーパスの誤りとして検出してしまう．

4.2 京大コーパスでの誤り検出

コーパスとして，京大コーパス version 2.0 の中から，1 月 1 日と 1 月 3 日から 9 日までの記事 9,204 文 (229,816 形態素) を使用した．

実際にコーパスの誤り修正の作業を行なう場合，誤りの検出と人手による修正作業を繰り返して行なうことで，始めのうちは検出されなかった誤りが検出できる可能性がある．そこで，コーパスの誤り検出を繰り返し行なった場合の振舞いを調べるために，本手法による誤り検出と人手による修正作業を繰り返して行なった結果を表 3 に示す．3 回目までのどの回も，誤りとして検出された要素は全て実際にコーパスの誤りだった．誤り検出と人手による修正を繰り返すことで，検出される誤りの数は急速に減っていき 4 回目には誤りは検出されなかった．結局，手作業による誤り修正のフィードバックを行なっても，別の新たな誤りはあまり検出されなかった．

表 3: 京大コーパス上で検出された誤りの個数

繰り返し回数	1	2	3	4
誤って検出された数	0	0	0	0
品詞付与の誤り数	64	9	2	0
形態素分割の誤り数	21	2	0	0
合計	85	11	2	0

4.3 WSJ コーパスでの誤り検出

コーパスとして，Penn Treebank WSJ コーパスを使用した．このデータは 53,113 文 (1,284,792 トークン) を含んでいる．

コーパスの誤り検出の結果，1,740 個の要素が検出された．

誤りとして検出された要素の中で，コーパスの始めから 200 個のものについて本当の誤りかどうかを人手で調べた．その結果，199 個が実際の誤りで 1 個が検出ミスだった (99.5%の精度)．表 4 が検出された誤りの例である．

5 関連研究

Abney らはブースティングを用いてコーパスの誤り検出を行なっている [1]．ブースティングは，SVM と同様に学習の難しい事例に対して大きな重みを付与する．彼らは，品詞タグと PP attachment 情報の誤り検出に応用して，うまく誤りを検出できることを示したが，具体的な精度などは報告されていない．また，新納は決定リストとアダブーストの組み合わせによるコーパスの誤り検出手法を提案している [5]．

コーパスの誤り検出について，確率に基づいた手法がいくつか提案されている [6, 4, 2]．村田らは，「修正前のタグが誤っている確率」を決定リストや用例ベース手法を用いて計算し，その値がしきい値以上のものを誤りとみなすという方法により 70〜80%の精度でコーパス中の形態素の誤り検出ができることを示した [7]．このような確率に基づいた方法は，一般性があり，修正後のタグが誤っている確率も扱うことによって誤りの検出だけでなく誤りの訂正も容易行なえるという利点がある．しかし，修正前のタグが誤っている確率がしきい値以下の事例は検出することができない．他方，我々の方法は確率は扱わずにコーパス中の非一貫性を検出するというアプローチであり，事例の出現頻度に関わらず誤りの検出を行なうことができる．また，実験では 90%以上という高い精度を得ることができた．しかしながら，我々の手法はコーパスの誤り自動訂正へ簡単に応用することはできない．

表 2: RWCP コーパス上で正しく検出された例と誤って検出された例

正しく検出された例	
する | こと | が | 確実 |に/助詞-副詞化| なっ| た 大阪 | 市 |中央/名詞-固有名詞-地域-一般| 区 | の レジャー | 用 |多目的/名詞-一般| 車	対決 | が | 確実 |に/助詞-格助詞-一般| なっ| た 大阪 | 市 |中央/名詞-一般| 区 | の レジャー | 用 | 多/接頭詞-名詞接続 |目的/名詞-一般| 車
誤って検出された例	
容疑 | 者 |と/助詞-並立助詞| 二 | 人 | の | 間 悪玉 | だ | など | と |いう/動詞-自立| の | は なろ | う | と | し |て/助詞-接続助詞| いる |。	容疑 | 者 |と/助詞-格助詞-一般| 二 | 人 | で 飲酒 | を | 勧める | など |という/助詞-格助詞-連語| の | は 標準 | 仕様 |として/助詞-格助詞-連語| いる |。

表 4: WSJ コーパス上で正しく検出された例と誤って検出された例

正しく検出された例	
, president and chief/JJ executive officer of New York Stock Exchange composite/JJ trading yesterday loss for its fiscal first quarter ended/VBN Sept. 30 .	named president and chief/NN executive officer of New York Stock Exchange composite/NN trading yesterday Revenue for its first quarter ended/VBD Sept. 30 was
誤って検出された例	
EOS 3/LS . EOS Send your child to a university .	Nov. 1-Dec . EOS 3/CD . EOS

6 まとめ

本稿では，SVM を用いて例外的な事例を検出する方法を考え，コーパス中から非一貫性を検出する方法を提案した．実験を行なった結果，90%以上の高い精度を得た．結果の評価は人手で行なっており，実験条件も異なるため他手法と単純に比較はできないが，これは人手によるコーパス修正作業の支援には，実用的な精度であると考えられる．しかしながら，コーパス中の全ての誤りに対してどれだけ検出できたかという再現率の評価は難しく，どれだけ多くの誤りを検出できるかという問題は今後の課題である．

本稿では，まず SVM を用いて例外的な事例の検出を行ない，その事例と共にコーパス中の一貫性を乱す箇所を抽出する方法を示した．しかし，SVM による学習は行なわなくても，直接 SVM の訓練データから，似た素性を持ってラベルの異なった事例の対を取り出すことで，非一貫性の抽出によるコーパスの誤り検出は行なえると考えられる．

参考文献

[1] Abney, S., Schapire, R. E. and Singer, Y.: Boosting Applied to Tagging and PP Attachment, *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pp. 38–45 (1999).

[2] Eskin, E.: Detecting Errors within a Corpus using Anomaly Detection, *Proceedings of the 6th Applied Natural Language Processing Conference and the 1st Meeting of the North American Chapter of the Association of Computational Linguistics*, pp. 148–153 (2000).

[3] Vapnik, V.: *Statistical Learning Theory*, Springer (1998).

[4] 乾孝司, 乾健太郎: 統計的部分係り受け解析における係り受け確率の利用法—コーパス中の構文タグ誤りの検出—, 情報処理学会研究報告 99-NL-134, pp. 15–22 (1999).

[5] 新納浩幸: 決定リストとアダブーストを利用した訓練データ中の誤り検出, 言語処理学会 第 7 回年次大会 発表論文集, pp. 26–29 (2001).

[6] 内山将夫: 形態素解析結果の誤りを発見する統計的尺度, 情報処理学会研究報告 99-NL-129, pp. 71–78 (1999).

[7] 村田真樹, 内山将夫, 内元清貴, 馬青, 井佐原均: 決定リスト, 用例ベース手法を用いたコーパス誤り検出・誤り訂正, 情報処理学会研究報告 2000-NL-136, pp. 49–56 (2000).

[8] 中川哲治, 工藤拓, 松本裕治: 修正学習法による形態素解析, 情報処理学会研究報告 2001-NL-146, pp. 1–8 (2001).