自然言語処理 131－10
音声言語情報処理 26－10
（1999．5．28）

ESCに基づく確率的決定リストを用いたテキスト分類

李 航　山西 健司

NEC C&Cメディア研究所

〒216-8555 川崎市宮前区宮崎4-1-1

{lihang,yamanisi}@ccm.cl.nec.co.jp

本稿では、確率的決定リストを用いたテキスト分類の一方法を提案する。この方法は、拡張型確率的コンプレキシティ(ESC)の最小化を基準に確率的決定リストを構築することを特徴とする。ロイター21578データによる評価実験では、その分類精度(break-even point)が82.0%に達することがわかった。これは同ベンチマークデータに対するルールに基づく方法による分類結果の中で最高水準のものである。

Text Classification Using ESC-based Stochastic Decision List

Hang LI　Kenji YAMANISHI

C&C Media Res. Labs., NEC Corporation

4-1-1 Miyazaki Miyamae-ku Kawasaki, 216-8555 Japan

Abstract

We propose a new method of text classification using stochastic decision lists. Our method is unique in that decision lists are constructed based on the principle of minimizing Extended Stochastic Complexity (ESC). The accuracy of classification achieved by our method in terms of break-even point for the Reuters-21578 data is 82.0%, which appears better than or comparable to those of exiting rule-based methods.

1 はじめに

本稿では，ルールに基づくテキスト分類の手法について考える．この手法は以下の通りである．まず，予め幾つかのカテゴリを用意し，一部のテキストをそれらのカテゴリに分類する．次に，分類されたテキストを基に何らかのルールを学習し，学習したルールを基に新しいテキストを分類していく．ルールに基づくテキスト分類法は最初 (ADW94; CS96) 等によって用いられた．それには，使われる知識が人間に分かりやすく，容易に修正可能である利点がある．

本稿では，ルールに基づくテキスト分類の新しい方法を提案する．この方法では，IF-THEN タイプの確率的ルールの順序つきリスト，確率的決定リストを知識表現とする．

本稿で提案する方法の最大の特徴は，拡張型確率的コンプレキシティ (Extended Stochastic Complexity，或は ESC) という量の最小化を基準に決定リストを構築することである．決定リストの学習は主に成長と刈り込みという二つのステップからなる．

我々の方法は，他のルールに基づく方法より学習方式がシンプルである点と理論的な根拠に支えられている点で優れていると思われる．実際，ESC を学習の基準として用いる場合，期待分類誤りが既存のどの基準を用いる場合よりも少いことが理論的に証明されている (Yam98)．我々の方法による Reuters-21578 データの分類の精度 (break-even 点) が 0.820 で，これは従来のルールに基づく方法同等以上のものである．

また，我々は，知識が容易に修正可能である利点を実験で確かめることができた．具体的には，学習できた決定リストに対して人手による簡単な修正を加えた後，分類の精度 (適合率) をほぼ維持しながカバー率 (再現率) を増やすことができることを確認できた．さらに，ルールに基づく方法はカテゴリが特定的である場合分類精度が特に高いこともわかった．

2 確率的決定リストを用いたテキスト分類

2.1 方式

この方法では，テキストを二値ベクトル (一般的に多値の離散ベクトル) とみなす．学習では，二値ベクトルとラベルの組みを入力する．二値ベクトルはテキストを表し，ラベルはその属するカテゴリを表す．入力は以下のように表現できる．

$$(d_1, c_1), (d_2, c_2), \cdots, (d_m, c_m) \qquad (1)$$

$$d_i = (w_{i1}, w_{i2}, \cdots, w_{ik}) \quad (i = 1, \cdots, m)$$

d_i は二値ベクトルを，c_i はラベルを表す $(i = 1, \cdots, m)$．w_{ij} は 1 と 0 を値とし，それぞれ単語 w_j が i 番目のテキストに現れることと現れないことを表す $(j = 1, ..., k)$．各単語 w_j は属性と呼ばれ，また属性の連言は項と呼ばれる．

確率的決定リストとは，IF-THEN タイプの確率的ルールの順序つきリストのことである．各ルールは分類の条件，分類の決定，確率からなる．図 1 では，確率決定リストの例を示す．例えば，最初のルールでは，条件は「tariff」と「trade」という単語がテキストに現れることを表し，決定は ⟨trade⟩ というカテゴリに分類することを表し，確率は ⟨trade⟩ に属する確率が 0.8 であることを表す．

テキスト分類では，新しいテキストが決定リストのどのルールの条件に満足するかをリストの最初から順番に

tariff=1 & trade=1 → ⟨trade⟩ 0.8
deficit=1 & export=1 import=1 → ⟨trade⟩ 0.7
japanese=1 & car=1 → ⟨trade⟩ 0. 9
textile=1 & trade=1 → ⟨trade⟩ 0.9
protectionism=1 & trade=1 → ⟨trade⟩ 0.7
korea=1 & surplus=1 → ⟨trade⟩ 0.6
1 → ⟨not-trade⟩ 0.7

図 1: 確率的決定リストの例

チェックし，条件が満足されるルールがあれば，そのルールの決定に従ってテキストを該当するカテゴリに振り分けていく．決定リストの最後は常にデフォルトの分類決定を表すルールである．

X が属性の空間を，Y がラベル (カテゴリ) の集合を表すとする．決定リストの IF-THEN タイプのルールが X の一つの有限分割を形成する．有限分割の各セルに対して，確率的決定リストが Y の上の条件付き確率分布を与える (Yam92)．

2.2 利点

確率的決定リストをテキスト分類に用いる利点として，使われる知識が人間に分かりやすく，容易に修正可能であることが挙げられる．決定リストの分かりやすさは主にそれが属性空間の有限分割上に定義されたことによるものである．有限分割の各セルにおける要素を一様に扱うことが知識の分かりやすさに大きく寄与している．

2.3 有効性

確率的決定リストを用いたテキスト分類では，カテゴリが特定的である場合に (例えば，⟨wheat⟩ は特定的であり，⟨commodity⟩ はそうでない) 精度が特に高い傾向にある．

カテゴリが特定的，例えば ⟨wheat⟩ である場合，与えられたテキストに「wheat」，「ton」，「farmer」等数少ない単語が現れるかどうかのことでそのテキストがカテゴリに属するかどうかを判断することができる．その場合，決定リストを用いた方法は，知識表現が数少い単語に基づく有限分割によって定められているので，分類の精度も高い．

有限分割をさらに細かくすることによって，決定リストをカテゴリが一般的である場合にも有効に利用することが原理的に可能である．しかし，そうすると，決定リストの学習の処理時間が著しく増加し，方法自体が実用的でなくなることがある．

3 確率的決定リストの学習

確率的決定リストの学習の目的は新しいテキストに対する分類の精度を最大限に上げること (分類の誤りを最小限に抑えること) である．その際，ルールの選択の基準が最も重要な要素となる．我々の方法では，拡張型確率的コンプレキシティ (ESC) の最小化をルールの選択の基準として用いる．以下それを ESC 最小原理と呼ぶ．

理論的には，あらゆる可能な決定リストを構築し，その中からある基準 (例えば，ESC 最小) からみて最も良い決定リストを選択すればよい．現実的には，効率的なアルゴリズムを用いてそれを実現する必要がある．本稿では，決定リストを k- 決定リストに限定し，その効率的な学習アルゴリズムを提案する．k- 決定リストとは，

項における属性の数が k までに限定された k- 項からなる決定リストのことである．

ESC に基づく決定リストの学習アルゴリズム (DL-ESC と呼ぶ) は属性選択，成長と刈り込みという三つのステップからなる．

属性選択では，確率的コンプレキシティ (Stochastic Complexity，或は SC) に基づいて，与えられたカテゴリと深く関係する単語を集め，属性とする．このステップの処理の目的はこれに続く処理を効率的にすることである．成長では，ESC 最小原理に基づいてルールを選択し，決定リストに順番に追加していく．このように得られる決定リストが学習データに過度に適合することがあるので，次の刈り込みでは決定リストの最後からルールを一つずつ刈り込んでいく．実際，ESC 最小原理の観点からみてこれ以上刈り込まないほうがよいところまで刈り込みを続ける．

本稿では，簡単のため，カテゴリが二つしかない場合のアルゴリズムについて説明する．カテゴリがそれ以上の場合への拡張は容易に行なえる．

3.1 SC と ESC

SC は与えられたデータに含まれる，ある確率モデルに対する情報の量を表す尺度である (Ris96; Ris97)．MDL 原理 (記述長最小原理) は，データの SC の最も小さいモデルがそのデータを生起した確率分布に最も近く，統計的推定ではそのモデルを選択すべきであると主張する．SC は確率モデルによるデータを記述するための最短符号長 (或は，記述長) としても解釈できる．

統計的決定理論の立場からみれば，SC は，モデルが確率分布で，損失関数が対数損失である条件の下で定義されたものである．最近，Yamanishi は SC を ESC に一般化した．ESC は，やはり与えられたデータに含まれる，あるモデルに対する量であるが，モデルが確率分布だけでなく任意の実数値関数のパラメトリック・クラスであってよい，かつ損失関数が対数損失だけでなく任意の歪み関数であってよいという意味で SC の拡張になっている．また，ESC 最小原理は，与えられたデータの，与えられた損失関数に対する，ESC の値が最も小さいモデルが未知のデータに対するその損失関数による予測誤りが最も少いと主張する．

MDL 原理が統計的推定のためのモデル選択基準としてよい性質をもつことが理論的に証明されている (Ris96; BC91; Yam92)．しかし，多くの分類問題では，モデル選択の目的は「真」のモデルを正しく推定することではなく，未知のデータに対する，ある損失関数による予測 (分類) 誤りを最少にすることである．その際，ESC 最小原理は MDL 原理より適当な基準となる．現に，ESC を学習の基準として用いる場合，一般損失関数に対する，期待予測 (分類) 誤りが既存のどの基準を用いる場合よりも少いことが理論的に証明されている (Yam98)．

3.2 属性選択

与えられたカテゴリに対して，そのカテゴリと深く関係する単語を集め，属性とする．具体的にはその単語の出現を考慮しない場合の SC と考慮する場合の SC の差を計算し，SC の差分が最も大きい単語を集める．

$(d_1, c_1), (d_2, c_2), \cdots, (d_m, c_m)$ は学習データであるとする．ここでは，d_i は i 番目のテキストを，c_i は i 番目のカテゴリ (ラベル) を表すとする $(i = 1, \cdots, m)$．さらに，$c^m = c_1 \cdots c_m$ と $d^m = d_1 \cdots d_m$ とする．$c_i \in \{0, 1\}$．$c_i = 1$ は d_i が与えられたカテゴリに属することを意味し，$c_i = 0$ はそうでないことを意味する．

(Ris96) によれば，ラベル c^m の SC を以下のように計算することができる．

$$SC(c^m) = mH\left(\frac{m^+}{m}\right) + \frac{1}{2}\log\frac{m}{2\pi} + \log\pi$$

ここでは，m^+ は c^m 中で値が 1 であるラベルの数を表す．log は自然対数を表すとする．さらに，

$$H(z) \stackrel{\text{def}}{=} -z\log z - (1-z)\log(1-z)$$

とする．

c^{m_w} は，対応するテキスト d_i の中に単語 w が現れる $c_i (c_i \in c^m)$ からなるラベル列であるとする．ここに，m_w は c^{m_w} におけるラベルの数であるとする．すると，c^{m_w} の SC の値を以下のように計算することができる．

$$SC(c^{m_w}) = m_w H\left(\frac{m_w^+}{m_w}\right) + \frac{1}{2}\log\frac{m_w}{2\pi} + \log\pi$$

ここでは，m_w^+ は c^{m_w} における値が 1 であるラベルの数を表す．一方，$c^{m_{\neg w}}$ は対応するテキスト d_i に単語 w が現れない $c_i (c_i \in c^m)$ からなるラベル列であるとする．$m_{\neg w}$ は $c^{m_{\neg w}}$ におけるラベルの数である．$c^{m_{\neg w}}$ の SC の値を以下のように計算することができる．

$$SC(c^{m_{\neg w}}) = m_{\neg w} H\left(\frac{m_{\neg w}^+}{m_{\neg w}}\right) + \frac{1}{2}\log\frac{m_{\neg w}}{2\pi} + \log\pi$$

単語 w の出現を考慮しない場合の SC と考慮する場合の SC の差 $\delta SC(w)$ は以下のように計算される．

$$\begin{aligned}
&\delta SC(w) \qquad (2)\\
&= \frac{1}{m}\Big(SC(c^m) - (SC(c^{m_w}) + SC(c^{m_{\neg w}}))\Big)\\
&= \left[H\left(\frac{m^+}{m}\right) - \frac{m_w}{m}H\left(\frac{m_w^+}{m_w}\right) - \frac{m_{\neg w}}{m}H\left(\frac{m_{\neg w}^+}{m_{\neg w}}\right)\right]\\
&\quad - \left\{\frac{1}{2m}\log\frac{m_w m_{\neg w}\pi}{2m}\right\}
\end{aligned}$$

$\delta SC(w)$ の大きい w は与えられたカテゴリによく現れる，或はほとんど現れない単語である．それらの単語がそのカテゴリと深く関係するとみることができる．実際の属性選択では，δSC が与えられている閾値 τ 以上の単語を全て集め，属性とする．

式 (2) における $[\cdots]$ 部分はテキスト分類等で情報利得と呼ばれるもので，それを用いた属性選択が非常に有効であることが知られている (例，(YP97))．また，式 (2) における $\{\cdots\}$ の部分はデータ数が少い時情報利得が過大になる傾向を抑えることができる．

3.3 成長

図 2は成長のアルゴリズムを示す．まず，選ばれた属性を基に可能なすべての k- 項を定義し，その集合を T とする．次に，T の項の中から学習データに一度も現れないものを取り除く．また，空の決定リスト A を用意する．次に，ESC 最小原理に基づきルールを選択し，これを一つずつ A に追加していく．具体的には，毎回 ESC の値の減少分がもっとも大きいルールを追加する．

学習データは $(d_1, c_1), (d_2, c_2), \cdots, (d_m, c_m)$ であるとする．但し，d_i は i 番目のテキストを，c_i は i 番目のカテゴリ (ラベル) を表すとする $(i = 1, \cdots, m)$．また，$c^m = c_1 \cdots c_m$ と $d^m = d_1 \cdots d_m$ とする．(Yam98) によれ

ば，ラベル c^m の ESC の値は以下のように計算することができる．

$$ESC(c^m) = Loss(c^m) + \lambda\sqrt{m \log m} \quad (3)$$

ここでは，λ は正の定数を表し，$Loss(c^m)$ はデフォルトの分類を行なう際の誤りの数を表す．デフォルトの分類とは，例えば，すべてのラベルが 0 であると仮定することである．

t は T の中の項であるとする．c^{m_t} は対応するテキスト d_i において t が真になる $c_i(c_i \in c^m)$ からなるラベル列であるとする．ここに，m_t は c^{m_t} におけるラベルの数であるとする．$Loss(c^{m_t})$ は t による分類を行なう際の誤りの数であるとする．一方，$c^{m_{\neg t}}$ は対応するテキスト d_i において t が偽になる $c_i(c_i \in c^m)$ からなるラベル列であるとする．ここに，$m_{\neg t}$ は $c^{m_{\neg t}}$ におけるラベルの数であるとする．$\neg t$ は t の否定を表す．$Loss(c^{m_{\neg t}})$ は $\neg t$ による分類を行なう際の誤りの数であるとする．c^{m_t} と $c^{m_{\neg t}}$ の ESC の値をそれぞれ以下のように計算することができる．

$$ESC(c^{m_t}) = Loss(c^{m_t}) + \lambda\sqrt{m_t \log m_t} \quad (4)$$

$$ESC(c^{m_{\neg t}}) = Loss(c^{m_{\neg t}}) + \lambda\sqrt{m_{\neg t} \log m_{\neg t}} \quad (5)$$

項 t による分類を行なう場合，ESC の値の減少分 $\Delta ESC(t)$ は以下のようになる．

$$\begin{aligned}
&\Delta ESC(t) \quad (6)\\
&= ESC(c^m) - (ESC(c^{m_t}) + ESC(c^{m_{\neg t}}))\\
&= \left[Loss(C^m) - Loss(C^{m_t}) - Loss(C^{m_{\neg t}})\right]\\
&+ \left\{\lambda(\sqrt{m \log m} - \sqrt{m_t \log m_t} - \sqrt{m_{\neg t} \log m_{\neg t}})\right\}
\end{aligned}$$

$\Delta ESC(t)$ は二つの部分からなる．$[\cdots]$ 部分は分類誤りであり，$\{\cdots\}$ 部分は学習データに過度に適合することを防ぐ補正項である．

$T :=$ {k-term}, $A{:=}\emptyset$, $D = \{(d_1, c_1), ..., (d_m, c_m)\}$;
do while (T は空集合でない)
  $\Delta ESC(t)$ を計算
  $t^* = \arg\max_{t \in T}(\Delta ESC(t))$
  if $\Delta ESC(t^*) > 0$
    $(t^* \rightarrow c^*, P(c^*))$ を A の最後に追加
    t^* の条件を満足する d を D から削除
    $t \subset t^*$ を満足する t を T から削除
  else
    ループから抜け出す
A にデフォルトのルールを追加
A を出力

図 2: アルゴリズム: 成長

3.4 刈り込み

刈り込みでは，成長の出力をその入力とする．全 ESC の値が最小となる決定リストが見つかるまで，決定リストの最後のルールを一つずつ削除していく．以下全 ESC の値の計算法について述べる．

まず，ラベル c^m の決定リスト A に対する ESC の値を，A におけるすべての項 t に対する ESC の値の和として定義する．

$$ESC(c^m|A) = \sum_t ESC(c^{m_t})$$

但し，$ESC(c^{m_t})$ は式 (4) のように計算される．

次に，c^m と A の全 ESC 値を以下のように定義する．

$$\begin{aligned}
&ESC(c^m : A) \quad (7)\\
&= ESC(c^m|A) + \lambda' L(A)\\
&= \sum_t Loss(c^{m_t}) + \lambda \sum_t \sqrt{m_t \log m_t} + \lambda' L(A)
\end{aligned}$$

ここでは，λ' は正の定数を表し，式 (3)-(5) における定数 λ と異る値をとってもよいとする．$L(A)$ は決定リスト A を符号化する時の符号長である[1]．式 (7) 右辺の第 1 の項とそれ以外の項の間にはトレードオフの関係がある．例えば，第 1 の項が小さくなる (決定リストのデータへの適合がより良くなる) と，それ以外の項が大きくなる (決定リストがより複雑になる)．式 (7) における全 ESC の値が最小となる決定リストを選択することは即ちこのトレードオフ関係の上で最適なバランスを図ることである．

A は刈り込み前の決定リストを，A' は刈り込み後の決定リストを表すとする．$ESC(c^m : A) \leq ESC(c^m : A')$，即ち，$ESC(c^m|A') - ESC(c^m|A) \geq \lambda'(L(A) - L(A'))$ が成り立つ限り，刈り込み処理を続ける．刈り込みのアルゴリズムを図 3に示す．

$A{:=}$ 成長の出力
do while (A はデフォルトのルールだけ)
  A の最後のルールを刈り込み，A' とする
  if $ESC(c^m|A') - ESC(c^m|A) \geq \lambda'(L(A) - L(A'))$
    ループから抜け出す
  else
    $A := A'$
A を出力

図 3: アルゴリズム: 刈り込み

3.5 DL-SC

ESC の代わりに，SC に基づいて決定リストの成長と刈り込みを行なうことも考えられる．本稿では，そのアルゴリズムを DL-SC と呼ぶ．

4 従来方法

現在までに数多くのテキスト分類法が提案されている (例えば，(LR94; LSCP96; LY97; SSS98))．この節では，代表的な方法を幾つか紹介し，次の節では，本研究の方法とそれらの方法との比較を行なう．この中で，Ripper のみがルールに基づく方法である．

[1] 具体的には，$L(A) = \log T + \log(T-1) + \cdots + \log(T-i+1)$ と計算する．但し，T は可能な項の数で，i は A におけるルール数である (Yam92)．

表 1: データ・セットの概要

	データ 1	データ 2	データ 3	データ 4
内容	本文	本文	タイトル	タイトル
カテゴリ数	90	8	90	8
学習用テキスト数	9603	9603	9603	9603
テスト用テキスト数	3299	3299	3299	3299
単語種類数	21995	21995	8611	8611
テキスト内平均語数	70.2	70.2	6.4	6.4

Rocchio

テキスト分類では，(Roc71) で提案された方法が最も古く，また最もよく用いられている．ここでは，その基本形について述べる．まず，各単語 w_i のテキスト t における値 t_i を計算する：

$$t_i = \frac{f_i \times \log(m/m_i)}{\sqrt{\sum_{i=1}^{k}(f_i)^2 \times (\log(m/m_i))^2}} \quad (i = 1, \cdots, k)$$

ここでは，f_i は単語 w_i のテキスト t における出現度数で，m は学習用テキスト数で，m_i は学習用テキスト中 w_i の出現したテキストの数である．次に，ベクトル $\mathbf{t} = (t_1, \cdots, t_k)$ をテキスト t の特徴ベクトルとする．また，ベクトル $\mathbf{c}$ をカテゴリ c の特徴ベクトルとする．

$$\mathbf{c} = \frac{1}{|c|}\sum_{t \in c}\mathbf{t}$$

分類では，新しいテキスト d の特徴ベクトル $\mathbf{d}$ と各カテゴリの特徴ベクトルとのコサイン値を計算し，コサイン値の最も大きいカテゴリに d を振り分ける．

BIM

Binary Independent Model (BIM) では，テキストを二値ベクトルと見なす (RJ76)．分類では，各カテゴリ c の新しいテキスト d に対する事後確率 $P(c|d)$ を計算し，事後確率のもっとも高いカテゴリに d を分類する．事後確率は，ベクトルにおける属性が互いに独立であるという仮定の下で，ベイズの定理を用いて計算する．

Naive Bayes

この方法では，カテゴリがそれぞれヒストグラムによって表現される確率分布をもつとする (KW78)．また，カテゴリに属するテキストがそのカテゴリの分布に従って独立に生成されたものであると仮定する．分類では，各カテゴリ c の新しいテキスト d に対する事後確率 $P(c|d)$ を計算し，事後確率のもっとも高いカテゴリに d を分類する．事後確率は，ベイズの定理をつかって計算する．

Ripper

(CS96) では，ルールに基づくテキスト分類法が提案されている．その学習で Ripper というアルゴリズムを用いる．Ripper の知識表現は確率的決定リストのとほぼ同様なものである．しかし，Ripper 自身は DL-ESC と異るアルゴリズムである．まず，Ripper では，属性選択は成長の各ステップにおいて行なわれる．それに対して，DL-ESC では，属性選択は成長の前に行なわれる．次に，Ripper では，成長と刈り込みは MDL 原理に基づいて行なわれる．それに対して，DL-ESC では，成長と刈り込みは ESC 最小原理に基づいて行なわれる．(しかし，すでに述べたように，分類の誤りを最少にすることを目的とするテキスト分類では，MDL 原理を用いるより ESC 最小原理を用いたほうが妥当である)．また，Ripper では，最後に最適化という処理を行なう．DL-ESC では同様の処理を行なわない．

従って，DL-ESC は Ripper よりずっとシンプルであり，DL-SC は Ripper の簡略版と見なすことができる．

5 実験結果

5.1 データと評価基準

ロイター通信新聞記事データ Reuters-21578[2] を用いて評価実験を行なった．まず，記事における単語の活用形を Oxford Learner's Dictionary [3] に従って原型に変換した．また，我々が作成した不用語辞書に基づいて不要語 (例えば，「the」) を記事から取り除いた．次に，Apte 分割に従って記事を学習用のものとテスト用のものに分けた．

次に，4つのデータ・セットを作成した．データ 1 では，各記事の本文をテキストとみなし，実験に用いる．また，Apte 分割の９０のトピック (例えば，「wheat」) をカテゴリとみなす．データ 2 では，９０のトピックの８つのメタトピック (例えば，「commodity」) をカテゴリと見なす．また，各記事の本文をテキストとして使う．データ 3 では，各記事のタイトルだけをテキストとみなし，実験に用いる．また，９０のトピックをカテゴリと見なす．データ 4 では，８つのメタトピックとタイトルを使う．表 1は４つのデータ・セットの概要を示す．

実験では，マイクロ平均による適合率 (precision) と再現率 (recall) を評価に用いた．この評価法 (LR94) では，適合率は分類できたテキスト中の正しく分類できたテキストの割合で，再現率は分類すべきテキスト中の正しく分類できたテキストの割合である．さらに，一つの評価値として break-even 点を用いた．break-even 点とは適合率と再現率が等しくなる点で，その値が大きいほど分類の精度が良い．

5.2 実験手順

４つのデータ・セットを使って，DL-ESC，DL-SC，Bayes，BIM と Rocchio による分類実験を行なった．

DL-ESC と DL-SC では，k- 決定リストの k を３に固定した．また，属性選択の閾値 τ を 0.001 に固定した．各カテゴリに対して，成長アルゴリズム ($\lambda = 0.1$) をつかって ESC に基づく決定リストを作成した．その後，刈り込みアルゴリズムを用いて決定リストの刈り込みを行なった．異る λ' の値 0, 1, 2 によって得られた異る決定リストを用いてテキスト分類を行ない，再現率 - 適合

[2] http://www.research.att.com/lewis.

[3] ftp://sable.ox.ac.uk.

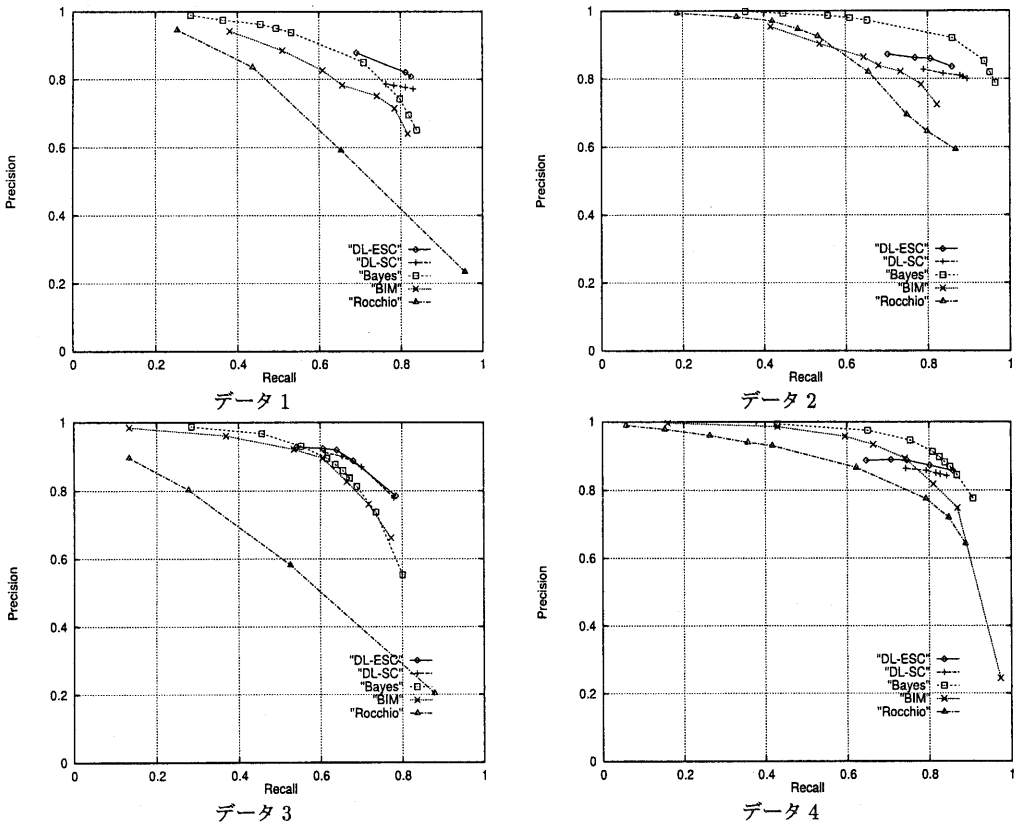

図 4: 各方法の再現率 - 適合率曲線

表 2: 各方法の break-even 点

	データ 1	データ 2	データ 3	データ 4
DL-ESC	.820	.843	.785	.858
DL-SC	.783	.820	.780	.843
Bayes	.773	.892	.736	.859
BIM	.747	.784	.733	.814
Rocchio	.625	.726	.553	.781

率曲線を作成した．DL-SC に対しても同様な実験を行なった．

DL-ESC の属性選択法を BIM にも適用した．しかし，異るデータ・セットに対して，異る閾値 τ を用いた．Bayes と Rocchio では，すべての単語を属性とした．なぜなら，そうしたほうがそれぞれの方法の精度が上げられることが予備実験でわかったからである．

図 4は各データ・セットに対する各方法の再現率 - 適合率曲線を示す．表 2は break-even 点を示す．

分類精度 (break-even 点) の意味で DL-ESC は常に DL-SC よりよいことがわかった．また，DL-ESC は常に BIM と Rocchio より精度がよいこともわかった．データ 1 と 3 では，DL-ESC は Bayes よりよく，データ 2 と 4 では，Bayes は DL-ESC よりよいこともわかった．

5.3 考察

DL-ESC と DL-SC

DL-ESC が常に DL-SC よりよい理由について考察した．結論として，テキスト分類では MDL 原理を用いるより ESC 最小原理を用いたほうがよいことがわかった．図 5はカテゴリが (gold) である場合のルール選択の状況を示す．単語「gold」と「ounce」が共に現れた学習用テキストの中で，63 個のテキストが実際カテゴリ (gold) に属し，9 個のテキストがそれに属さない．単語「gold」が現れたテキストの中で，94 個のテキストが実際カテゴリ (gold) に属し，90 個のテキストがそれに属さない．明らかに，項「gold=1 & ounce=1」からなるルールを選択したほうが分類の誤りが少い (精度がよい)．ESC に基づいてルールを選択する場合，確かに前者の項が選ばれる．というのは，その項による分類の ESC の減少分が大きいからである．しかし，SC に基づいてルールを選択する場合，前者の項が選ばれず，後者の項が選ばれる (表 3を参照)．

表 4は幾つかのカテゴリにおける ESC に基づく決定リストの最初のルールと SC に基づく決定リストの最初のルールを示す．ESC に基づく決定リストがより細かいルールをもっていることがわかる．これにより ESC に

表 4: 決定リストの最初のルール

DL-ESC	DL-SC
acquire=1 & cts=0 & rate=0 → ⟨acq⟩	acquire=1 → ⟨acq⟩
wheat=1 & lt=0 → ⟨grain⟩	wheat=1 → ⟨grain⟩
beef=1 & lt=0 → ⟨carcass⟩	beef=1 → ⟨carcass⟩
money=1 & supply=1 & lt=0 → ⟨money-supply⟩	money=1 & supply=1 → ⟨money-supply⟩
wheat=1 & ton=1 → ⟨wheat⟩	wheat=1 → ⟨wheat⟩

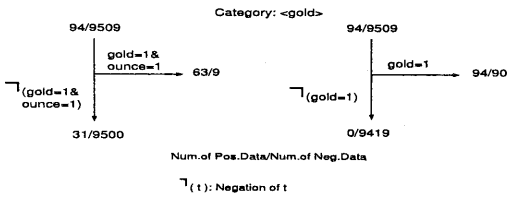

図 5: 学習データの例

表 3: $\Delta ESC(t)$ の値 ($\lambda = 1$) と $\Delta SC(t)$ の値

	gold=1 & ounce=1	gold=1
ΔESC	37.69	−23.81
ΔSC	0.03	0.04

表 5: ルールに基づく方法の結果

方法	break-even 点	文献
C4.5	.794	(Joa98)
BayesNets	.800	(DPHS98)
Ripper	.820	(CS98)
DL-SC	.783	本稿
DL-ESC	.820	本稿

表 6: 一般化の例

カテゴリ	属性	集合属性
alum	aluminium	{aluminium, aluminum}
cpi	february	{january,..., december}
money-fx	stg	{stg,dollar,yen}
trade	deficit	{surplus, deficit}
trade	import	{import, export}

基づく決定リストがより高い分類精度をもつ.

DL-ESC と Bayes 等

Rocchio, BIM と Bayes はルールに基づく方法ではない. それらの方法では, 知識表現は直接属性空間上に定義される. 一方, DL-ESC では, 知識表現 (決定リスト) は属性空間の有限分割上に定義される. また, そのことが知識のが可読性と修正しやすさとつながっていく.

DL-ESC が常に BIM と Rocchio より精度がよいことは, 属性空間の有限分割上の知識表現でも適切な学習法を用いれば, 高精度の分類を実現できることを意味する.

また, カテゴリが特定的である場合, DL-ESC が Bayes より精度よく, カテゴリが一般的である場合, Bayes が DL-ESC より精度がよいこともわかった. これは, 確率的決定リストを用いた方法はカテゴリが特定的である場合により有効であることを意味する.

従来のルールに基づく方法との比較

表 5では, 報告される, Reuters-21578 データ (データ 1) に対する他の広い意味でのルールに基づく方法の分類精度 (break-even 点) を示す. 本研究の方法は, 従来のルールに基づくに方法と同等以上の精度を有することがわかる. 特に, DL-ESC は Ripper よりかなりシンプルであるにも関わらず, それと同等の精度をもつことが興味深い.

SVM との比較

最近, Support Vector Machine (SVM) によるテキスト分類法が提案され, 高い精度をもつことが報告されている (break-even 点は 87.0%) (Joa98; DPHS98). SVM の知識表現は直接属性空間上に定義されるもので, 決定リストの知識表現とはかなり異る. そのことが SVM が決定リストより高い分類精度を達成できる一つの原因になっていると思われる. しかし, 逆にそのことによって知識が人間に分かりにくくなり, また修正も困難となっている.

5.4 決定リストの後修正

ルールに基づく方法は, それ以外の方法に較べて知識が人間に分かりやすく修正しやすい等の利点をもつ. それを確認するため以下の実験を行なった.

上記実験で得られた決定リストの中に明らかに人手による修正を加えたほうがよいものがあった. 実際, $\lambda = 0.1$ の条件の下でデータ 1 から学習した決定リストのいくつかに修正を加えた. 具体的には, ルールに現れた幾つかの属性をその同義語, 反対語, 或は関連語からなる集合属性に一般化した. 例えば, 属性「Feburary」を「January」... 「December」からなる集合属性に一般化した. 集合属性の中の任意の一つの属性の値が 1 となれば, その集合属性の値が 1 となるとする. 表 6で他の一般化の例を示す. 表 7では, このような修正前後の幾つかのカテゴリに対する分類結果を示す.

一部のカテゴリに対して, 人手による修正によって, 適合率をほぼ維持しながら再現率を大きく上げることができることがわかった. これは, ルールに基づく方法以外の方法のもち得ない利点である. 現実の応用では, 相対的に少い学習データから知識を学習し, 実世界における無限の「テスト・データ」に対して学習した知識を適用する必要がある. その場合, 学習した知識が容易に修正可能ということが極めて重要な利点であると思われる.

表 7: 修正前後の分類結果

カテゴリ	修正前		修正後	
	再現率	適合率	再現率	適合率
alum	.478	1.00	.652	1.00
cpi	.286	.727	.429	.706
money-fx	.559	.621	.581	.608
trade	.744	.531	.803	.527
平均	.517	.719	.616	.711

6 おわりに

本稿では，ルールに基づくテキスト分類の一方法を提案した．ルールに基づく方法は，知識が人間に分かりやすいなどの利点をもつので，今後広く利用されていくと思われる．

本研究の提案する方法は，確率的決定リストを知識表現とし，ESC 最小原理を学習の基準とする．学習方式がシンプルであることと理論的な根拠に支えられていることが本研究の方法が他のルールに基づく方法より優れた点である．本研究の実験結果によれば，この方法は，従来のルールに基づく方法と同等以上の精度を有することもわかった．

また，本稿では，知識が容易に修正可能であるというルールに基づく方法の利点を実証した．さらに，ルールに基づく方法はカテゴリが特定的である場合に分類精度が特に高いことをも実証した．

本稿で提案する方法はテキスト分類に限らず，データ・マイニングや自然言語処理など幅広い分野で利用できる．

参考文献

Chidanand Apte, Fred Damerau, and Sholom M. Weiss. Automated learning of decision rules for text categorization. *ACM Trans. on Information Systems*, 12(3):233–251, 1994.

Andrew R. Barron and Thomas M. Cover. Minimum complexity density estimation. *IEEE Transaction on Information Theory*, 37(4):1034–1054, 1991.

William W. Cohen and Yoram Singer. Context-sensitive learning methods for text categorization. *Proc. of SIGIR'96*, 1996.

William W. Cohen and Yoram Singer. Context-sensitive learning methods for text categorization. *http://www.research.att.com/singer*, 1998.

Susan Dumais, John Platt, David Heckerman, and Mehran Sahami. Inductive learning algorithms and representations for text categorization. *Proc. of ECML'98*, 1998.

Thorsten Joachims. Text categorization with support vector machines: Learning with many relevant features. *Proc. of ECML'98*, 1998.

Gautam Kar and Lee J. White. A distance measure for automatic document classification by sequential analysis. *Information Processing and Management*, 14:57–69, 1978.

David D. Lewis and Marc Ringuette. A comparison of two learning algorithms for test categorization. *Proc. of 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 81–93, 1994.

David D. Lewis, Robert E. Schapire, James P. Callan, and Ron Papka. Training algorithms for linear text classifiers. *Proc. of SIGIR'96*, 1996.

Hang Li and Kenji Yamanishi. Document classification using a finite mixture model. *Proc. of ACL'97*, pages 39–47, 1997.

Jorma Rissanen. Fisher information and stochastic complexity. *IEEE Transaction on Information Theory*, 42(1):40–47, 1996.

Jorma Rissanen. Stochastic complexity in learning. *Journal of Computer and System Sciences*, 55:89–95, 1997.

S.E. Robertson and K. Sparck Jones. Relevance weighting of search terms. *Journ. of the American Society for Information Science*, 27:129–146, 1976.

J. Rocchio. Relevance feedback information retrieval. In Gerard Slaton, editor, *The Smart Retrieval System -Experiments in Automatic Document Processing*, pages 313–323. Prentice-Hall, 1971.

Robert E. Schapire, Yoram Singer, and Amit Singhal. Boosting and rocchio applied to text filtering. *Proc. of SIGIR'98*, 1998.

Kenji Yamanishi. A learning criterion for stochastic rules. *Machine Learning*, 9:165–203, 1992.

Kenji Yamanishi. A decision-theoretic extension of stochastic complexity and its applications to learning. *IEEE Trans. on Inf. Thy.*, 44(4):1424–1439, 1998.

Yiming Yang and Jan O. Pedersen. A comparative study on feature selection in text categorization. *Proc. of ICML'97*, pages 412–420, 1997.