CA-003

q 元 Exhaustive Code を用いた多値分類手法に関する研究
A multi-value classification method using q-ary Exhaustive Code

雲居玄道 * 八木秀樹 † 小林学 * 後藤正幸 * 平澤茂一 *

Gendo Kumoi Hideki Yagi Manabu Kobayashi Masayuki Goto Shigeichi Hirasawa

1 はじめに

近年，情報化社会の到来により，World Wide Web，電子メール，電子図書館など，膨大なオンラインテキストが扱われるようになった．このような電子媒体のテキストデータを自動処理する技術の重要性は高まる一方であり，中でも高精度な文書自動分類技術が必要とされている．

文書の自動分類技術には様々な手法が提案され，それらの多くは分類対象となるカテゴリ数が M (≥ 3) となるような多値分類問題である．そのことから，多値分類器の構成法について様々な観点から盛んに研究されている．この多値分類器の構成法には大きく分けて 2 つのアプローチがある．1 つは直接，多値分類器を構成する方法であり，SVM を多値分類へ拡張した手法 [1] や Random Forest [2] などがある．もう 1 つのアプローチとして，二値判別器を組合せて多値分類器を構成する方法がある．後者は，多値分類器を直接構築した際の精度低下を，性能の高い二値判別器の組合せによって解決できることから注目されてきた．また，前者の手法においては，近年の PC 性能の大幅な向上により，計算コストをかけることが可能となり，分類性能が向上してきている．

二値判別器を組合せて多値分類器を構成する手法としては，符号理論の枠組を導入した Error-Correcting Output Codes（ECOC）法に基づく多値判別法 [4]–[6] が知られている．ECOC 法は，N 個の 2 値判別器の次元で構成される M 個のカテゴリに対応する “符号語” で構成した “符号語表” を用いて，2 値判別器の出力結果全体から符号語を推定するものである．これにより，2 値判別器のいくつかで誤判別が生じたとしてもカテゴリを推定することが可能となる．この ECOC 法に基づく多値判別法に，符号語表の構成法に関する研究 [3, 4, 5, 6] がある．この符号語表の構成法においては，符号語表の要素を $\{1, 0\}$ の 2 元として各 2 値判別器においてすべてのカテゴリを正例または負例に判別する手法 [3, 4, 5] と $\{1, 0, *\}$ の 3 元として，2 値判別器によっては正例にも負例にも判別しないカテゴリの存在を許容した符号語表を用いる手法 [5, 6] の 2 種類がある．また，2 元符号語表の代表的なものに，全ての 2 値判別器から構成される含む Exhaustive Code [4] が提案されている．また，$\{1, 0, *\}$ の 3 元の全ての構成法として Exhaustive Ternary Codes [6] が提案されている．

これらの手法は，2 値判別器を組合せて多値分類器を構成する手法である．これにより，多値分類問題を 2 値分類という部分問題に落とし込むことにより，性能が向上することを示してきた．この背景には，強力な 2 値判別器である SVM を活用したいというモチベーションと共に，直接的に M 値の多値判別器を構成する場合には，分類問題が複雑となり性能が低下することがあげられる．一方で，M 値分類器と 2 値判別器の間には，3 値や 4 値などといった多値判別器も存在している．そのため，これらの部分問題を考慮することにより性能が向上することが考えられる．

そこで，本研究においては，M 個のカテゴリの分類問題を 2 値判別器に加えて，3 値，4 値と M 値までの多値判別器の組合せで多値分類器を構成することを考える．具体的には，従来の 2 元 Exhaustive Code を拡張した，q 元 Exhaustive Code を提案する．この提案する符号語表について，符号長，および符号語表の構成アルゴリズムについて述べる．そして，ベンチマークデータを用いて，提案する指標の有効性を検証する．

2 2 元 Exhaustive Code

Dietterich と Bakiri が提案した Exhaustive Code [4] の符号語構成法では，構成可能な全ての 2 値判別器の組合せを表現した符号語表である．このとき，例えば各 2 値判別において，$(0, 0, 1)$ と $(1, 1, 0)$ のように 0,1 を反転させたものは，分類器としては同じものになることに注意が必要である．そのため，判別器数である符号長は，2^M の全ての組合せから，全てが 0 または 1 と同値となるものを 2 個取り除いた上で，反転したものが同じであるため，2 で除算したものとなる．す

* 早稲田大学
† 電気通信大学

表 1: 符号長 N（M=8）

q	2	3	4	5	6	7	8
$N(M,q)$	127	1,093	2,794	3,844	4,110	4,138	4,139
$N_{\mathrm{Only}}(M,q)$	127	966	1,701	1,050	266	28	1

表 2: 符号長 N（M=20）

q	2	3	4	17	18	19	20
$N_{\mathrm{Only}}(M,q)$	524,287	580,606,446	45,232,115,901	741,284	15,675	190	1

なわち，符号長 N は，

$$N = \frac{2^M - 2}{2} = 2^{M-1} - 1 \tag{1}$$

となる．

3 q 元 Exhaustive Code

2 元 Exhaustive Code は，全ての 2 値判別器の組合せにより構築されていた．本研究では，これを多値判別器に拡張する．多値判別器への拡張を考えた時，カテゴリ数 M に対して，$M \geq q \geq 2$ 値の判別器が考えられる．そこで，全ての q 値分類器の組合せで多値分類器を構成する q 元 Exhaustive Code を提案する．

この際，q 元 Exhaustive Code と呼称する場合に，$2, 3, \cdots, q$ 元までの全てを含んだ構成法と q 元のみの構成法の 2 つが考えられる．また，q の値域はカテゴリ数 M に依存する．そこで，前者を (M,q) Exhaustive Code，後者を Only (M,q) Exhaustive Code と記述する．すなわち，Only (M,q) Exhaustive Code は，構成可能な全ての q 値の多値判別器が全て含まれるものであり，(M,q) Exhaustive Code は，構成可能な q 値以下の全ての多値判別器が全て含まれるものとなっている．なお，$q = 2$ の場合は，両 Exhaustive Code は従来の 2 元 Exhaustive Code と一致する．

3.1 Only (M,q) Exhaustive Code の符号長

Only (M,q) Exhaustive Code は，構成可能な全ての q 値の多値判別器の組合せからなる．この全ての組合せの数である，Only (M,q) Exhaustive Code の符号長 $N_{\mathrm{Only}}(M,q)$ は，

$$N_{\mathrm{Only}}(M,q) = \frac{l(M,q)}{q!}, \tag{2}$$

となる．ただし，

$$l(M,q) = \begin{cases} q^M - M - \sum_{r=2}^{q-1} \left(l(M,r) \times \binom{q}{r} \right), & M \geq q > 1 \\ 0 & \text{othewise.} \end{cases} \tag{3}$$

となる．

式 (2) にある $l(M,q)$ は，全ての q 値分類器の数を示している．このとき，2 元 Exhaustive Code と同様に，$0, 1, \cdots, q-1$ を入れ替えたものは同じ判別器となるため，その組合せ数である $q!$ で除算している．

また，その $l(M,q)$ は，式 (3) によって計算される．まず，M カテゴリにおいて，q 値の全ての組合せは，q^M となる．そこから，全てが同値であり分類が行われない M 個を取り除く．その上で，q 未満の全ての判別器を除く計算を第 3 項で行っている．例えば，$q = 3$ のときは，2 値判別器を取り除くことになるが，その際に，(0,1),(0,2),(1,2) の 3 種で構成された 2 値判別器がある．そのため，$\binom{q}{r}$ を積算している．そして，M カテゴリの分類問題において，$q > M$ の場合および $q \geq 1$ では，多値判別器が存在しないため，0 としている．

3.2 (M,q) Exhaustive Code の符号長

(M,q) Exhaustive Code の符号長 $N(M,q)$ は，Only (M,q) Exhaustive Code の符号長を加算して求めることができ，

$$N(M,q) = \sum_{r=2}^{q} N_{\mathrm{Only}}(M,q), \tag{4}$$

と表せる．表 1 に $M = 8$ の各 q に対する符号長，表 2 に $M = 20$ の一部の q に対する符号長を示す．

4 評価実験

本研究の多値判別器には Random Forest（以下，RF）[2] を用いる．RF のパラメータを表 5 に示す．

4.1 実験データ

実験データには，表 3 に示した 20 Newsgroups data set [7] の記事を用いる．また，このデータセットより 8 カテゴリを抽出し実験を行うため，抽出したカテゴリを表 4 に示す．

表 3: 実験データ

カテゴリ名（数）	rec.autos，rec.motorcycles，rec.sport.baseball，rec.sport.hockey，sci.crypt，sci.electronics，sci.med，sci.space，comp.graphics，comp.os.ms-windows.misc，comp.sys.ibm.pc.hardware，comp.sys.mac.hardware，comp.windows.x，misc.forsale，talk.politics.misc，talk.politics.guns，talk.politics.mideast，talk.religion.misc，alt.atheism，soc.religion.christian (20)
文書の特徴ベクトル	130,107 語
実験データ数	18,846 件
訓練データ	11,314 件
テストデータ	7,532 件

表 4: 8 カテゴリ実験データ

カテゴリ名（数）	rec.autos，rec.motorcycles，rec.sport.baseball，rec.sport.hockey，sci.crypt，sci.electronics，sci.med，sci.space (8)
実験データ数	7,931 件
訓練データ	4,762 件
テストデータ	3,169 件

表 5: Random Forest parameters

The number of trees in the forest	300
Criterion	Gini impurity
The number of features	$\lfloor\sqrt{130,107}\rfloor$

4.2 実験結果

表 4 の 8 カテゴリによる実験結果を表 6, 7 に，表 3 の 20 カテゴリによる実験結果を表 8 に示す．このとき，表 6, 8 において，$M = q$ となる実験結果は，RF を多値分類に適用した比較手法（従って符号長は 1）である．また，RF において木の数を変化させた結果を表 9 に示す．

表 6: Only (M, q) Exhaustive Code による実験結果 $(M = 8)$

q	$N_{\mathrm{Only}}(M, q)$	誤り率
2	127	0.080
3	966	0.079
4	1,701	0.079
5	1,050	0.080
6	266	0.080
7	28	0.080
8	1	0.103

5 結果のまとめと考察

本研究において，従来，2 値判別器に限られていた符号語表構成を多値判別器の構成に拡張する提案を行った．この結果として，表 6, 7 より，$q = 2$ である 2 元 Exhaustive Code を用いた従来手法と同等または改善された結果を示した．その中でも，Only (M, q) Exhaustive Code を適用した場合の $q = 4, 5$，および (M, q) Exhaustive Code を適用した場合の $q = 4, 5, 6$ において，従来の 2 元 Exhaustive Code よりも改善が見られた．これらは，符号長が長くなり，用いられる判別器の数が増えたことが原因であると考えられる．一方で，(M, q) Exhaustive Code において，$q > 6$ の場合には誤り率が悪化していることから，単純に組合せる判別器の数を増やせば，誤り率が改善されるわけではないこともわかる．

表 7: (M, q) Exhaustive Code による実験結果 $(M = 8)$

q	$N(M, q)$	誤り率
2	127	0.080
3	1,093	0.080
4	2,794	0.079
5	3,844	0.079
6	4,110	0.079
7	4,138	0.080
8	4,139	0.080

表 8: Only (M, q) Exhaustive Code による実験結果 $(M = 20)$

q	$N_{\mathrm{Only}}(M, q)$	誤り率
19	190	0.206
20	1	0.219

表 9: RF を用いた実験結果 ($M = 8$)

木の数	誤り率
300	0.103
300×28	0.100
300×127	0.098

以上のように，提案手法の分類性能面での有効性を示したが，有効な符号語表は，カテゴリ数 M に応じて指数的に増大するという問題がある．そのため，8 カテゴリ程度の多値分類問題には適用可能であるが，本研究で用いた 20 カテゴリのようなカテゴリ数が大きくなるところでは，現実的な適用が難しい．

そこで，注目すべきは，Only (M, q) Exhaustive Code における $q = M - 1$ の結果である．この符号語表においては，従来の 2 元 Exhaustive Code と同等の結果が得られている．2 元 Exhaustive Code は，符号長が $N_{\mathrm{Only}}(M, 2) = 2^{M-1} - 1$ と M によって指数的に増大する．これに対して，Only $(M, M-1)$ Exhaustive Code は，符号長が $N_{\mathrm{Only}}(M, M-1) = \frac{M(M-1)}{2}$ と M に対して 2 乗のオーダーで済むことから，効率的な符号語構成法といえる．

本研究で対象とした，20 Newsgroups data set の 20 カテゴリの分類問題を扱う際には，表 2 に示したように，Only $(20, 2)$ Exhaustive Code では，約 52 万，$(20, 20)$ Exhaustive Code に至っては，50 兆以上もの判別器が必要となる．これに対して，Only $(20, 19)$ Exhaustive Code では，符号長は 190 となり，現実的な時間で学習が可能となる．その結果を表 8 に示した．こちらでは比較手法のみとの比較であるが，誤り率が改善されていることが分かる．このことから，カテゴリ数が大きい場合に，提案手法である Only $(M, M-1)$ Exhaustive Code を用いることの有効性が示されている．

また，本研究では多値判別器として RF を用いた．この際の各 RF は木の数を 300 としたが，一般的に木の数を増やすことにより誤り率が改善することが知られている．そのため，8 カテゴリのケースにおいて木の数を増やしたが，表 9 から改善は限定的であることがわかった．

6 まとめと今後の課題

本研究において，提案手法の有効性を示した．特に，Only $(M, M-1)$ Exhaustive Code が従来手法である 2 元 Exhaustive Code と同等の性能を示したことで，カテゴリ数が増大した場合でも，現実的な学習時間のもとで，誤り率の低い分類が可能となった．

一方で，表 7 に示したように，(M, q) Exhaustive Code では $q > 6$ において，誤り率が増大する結果となった．これは，分類器が増える中において性能の低い分類器が増えることが問題だと考えられる．

筆者らは，2 元 Exhaustive Code に対して，用いる 2 値判別器を選択することで性能が改善されること示している [8]．本研究においても，$q = M$ とした全ての q 元 Exhaustive Code から多値判別器を選択して，部分集合を構成することで，よりよい多値分類器が構成可能と考えられ，今後の課題である．

参考文献

[1] Crammer, K. and Singer, K.: On The Algorithmic Implementation of Multiclass Kernel-based Vectormachines, *Journal of Machine Learning Research*, vol.2, pp.265–292 (2001).

[2] Leo, B.: Random Forests, *Machine learning*, vol.45(1), pp.5–32 (2001).

[3] 後藤正幸, 小林学.: 入門パターン認識と機械学習, コロナ社 (2014).

[4] Dietterich, T.G. and Bakiri, G.: Solving Multiclass Learning Problems Via Error-Correcting Output Codes, *Journal of Artificial Intelligence Research*, vol.2, pp.263–286 (1995)

[5] Escalera, S., Pujol, O. and Radeva, P.: Error-Correcting Output Codes Library, *Journal of Machine Learning Research*, vol.11, pp.661–664 (2010).

[6] Park, S. H., and Frnkranz, J.: Efficient Decoding of Ternary Error-Correcting Output Codes for Multiclass Classification, *In Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Springer, pp. 189–204 (2009).

[7] Lang, K.: Newsweeder: Learning to Filter Netnews, *Proceedings of the Twelfth International Conference on Machine Learning*, pp. 331–339 (1995).

[8] 雲居玄道，八木秀樹，後藤正幸，平澤茂一.: 符号理論の観点による二値判別器の相関に着目した多値文書分類のための符号語構成法, 情報処理学会 第 *115* 回数理モデル化と問題解決研究発表会, vol.2017-MPS-115(4), pp. 1–6 (2017)