情報システムと社会環境 83－4
（２００３．３．１４）

Support Vector Machineを用いた電子メールの自動分類

神奈川大学理学部情報科学科

米倉 正和† 堀 幸雄‡ 後藤 英一§

現在電子メールを狙った悪質な広告メールが社会問題になっている．その対策として主に単項条件式やパターンマッチングを用いたフィルタリングが用いられているが，これらの手法では既知の特徴を持つ電子メールしか分類できない．機械的学習による分類を加えることによって未知の電子メールに対しても分類性能を高めることができる．本研究では機械的学習としてSupport Vector Machine(SVM) を用い、電子メールの本文から特徴を抽出し学習させたところ，複雑な前処理無しに高い分類性能が出ることを確認した．

Filtering E-mail using Support Vector Machine

Department of Information Science,
Faculty of Science, Kanagawa University

Masakazu Yonekura† Yukio Hori‡ Eiichi Goto§

The wicked advertising mail which aimed at the present E-mail is a social problem. Although filtering using a unary condition formula and pattern matching as the measure is mainly used, only an E-mail with the known feature can be classified according to these techniques. A classification performance can be raised also to a strange E-mail by adding the classification. In this research, when the feature was made to extract and learn from the text of an E-mail, using Support Vector Machine (SVM) as machine learning, it checked that a classification performance high without a complicated pretreatment came out by machine learning.

1 はじめに

電子メールの利用はパソコンから携帯端末に広がり、電話と並ぶ重要なコミュニケーション手段となっている．しかし電子メールは大量送信時のコストの安さから悪質な広告業者の的になり、勝手に送り付けられる悪質な広告メール(以下『迷惑メール』と表記) が社会問題になっている．こうした迷惑メールへの対抗策として、電子メールを一定のルールに従って自動で分類するフィルタリングと呼ばれる手法が用いられる．

フィルタリングの中でも最も広く使われているものは人が決めたルールに基づいて分類を行うものである．この手法は分類対象が既知であり，限定されている場合に有効である．しかしそれぞれのルールに相関があるような場合は調節が困難になるため，様々な要素が絡み合うような複雑な分類には向いていない．そのため送り手の情報をデータベースと照合[8]したり，キーワードの有無を用いるといった，それぞれが独立した単純な分類規則の組み合わせが使われることになる．しかし迷惑メールはこうした分類の目安になるような『静的』な情報を改窮するなど手口が巧妙化し，単純な分類規則による分類は困難になってきている．またこの手法では，現在対応できていても分類対象が変化するとその都度新たにルールを作成したりパラメータの調整する必要が生じ，コストがかかってしまう．

こうした背景から2002年の7月に迷惑メールに対して規制を行う法律[1][2]の施行が行われ，受信者の同意なく商品やサービスの提供を宣伝するメールには件名の先頭に『未承諾広告※』の文字列を付けること等が義務付けられるようになった．これにより法を守った広告メールのフィルタリングは非常に容易になった．しかし，違法な広告メールや規制の及ばない海外からの迷惑メールは後を絶たない．

そこで本論文では機械的学習の１つであるSupport Vector Machine[4][5](以下『SVM』と表記) によるテキストの自動分類に注目し，電子メール本文を重視したフィルタリングを検証する．電子メール本文は人が決めたルールに基づいた分類方法ではその情報を用いて精度を上げることが難しく，それほど重視されてはこなかった．しかし，本来迷惑メールであるかどうかの判断は本文の内容によって判断されるのが自然であると考えられる．

機械的学習を文章 (テキスト) の分類に用いる際に問題となるのが学習数と特徴数の問題である．一般に機械的学習で特徴が多い複雑な分類をするにはそれ相応の学習データが必要になるが，文章の場合は学習数以上に特徴数が増えることがほとんどであるため，そのままでは満足な学習データを与えることが出来ない．そのため分類の際に重要と考えられる特徴を選び出して使用するという前処理が必要になる．しかしこの前処理はデータ全体の解析が必要となり，最適な値は実験的に求めるものであるため，コストの高い処理である．

しかし，機械的分類の中でも高い分類能力が報告されているSVMを用いて電子メールの自動分類を行ったところ，こうした複雑な特徴選択を必要とせずに十分に高い分類性能が出ることを確認した．

本論文の構成は次のようになっている．まず2章でSVMの理論的背景を述べる．3章でSVMの電子メール分類への適用法を説明する．4章で実際にSVMを用いて電子メールの分類を行い性能の評価を行い，5章でSVMを用いたフィルタリングシステムの例を示す．そして6章で論文を締めくくる．

2 Support Vector Machine

2.1 特徴空間の分割

全てのデータは正，負のどちらかのクラスに属するとする．l 個の学習データが与えられたとき，各学習データは学

†kazu@goto.info.kanagawa-u.ac.jp
‡hori@goto.info.kanagawa-u.ac.jp
§goto@goto.info.kanagawa-u.ac.jp

習データが持つ特徴ベクトル $\mathbf{x}_i$ と，学習データが属するクラスを示すスカラー値 y_i の組から成る．つまり各学習データは

$$(\mathbf{x}_i, y_i) \quad \text{ただし } \mathbf{x}_i \in \mathbf{R}^n \quad y_i \in \{-1, 1\}$$

として与えられる．
これらの学習データを特徴空間上で，ある超平面を境に分割する．この超平面を分離超平面と呼ぶ．分離超平面は

$$f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b$$

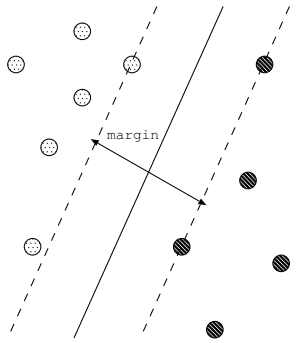
において，$f(\mathbf{x}) = 0$ となる点の集合とする．このような関数 f を識別関数と呼ぶ．

$$\text{sign}[\alpha] = \begin{cases} 1 & \text{if } \alpha \geq 0 \\ -1 & \text{if } \alpha < 0 \end{cases}$$

とする関数 sign を定義すると，未知のデータ $\mathbf{x}$ が属するクラス y は $y = \text{sign}[f(\mathbf{x})]$ で推定される．
機械的学習では学習データを元に，未知のデータに対して識別精度が高い識別関数 $f(\mathbf{x})$ を得ることが目的になる．

2.2 SVM の戦略

SVM は学習データを 2 つのクラスに分類するときに，学習データと分離超平面との間隔 (**マージン**) を最大化する戦略を用いた手法である．



マージンの最大化は汎化誤差最小化を意味する [5] ため，識別精度の向上が計れる．

2.3 SVM の手法

2.2 節で述べたマージンが最大になる分離超平面は次のように求められる．
学習パターンを分割する分離超平面

$$\mathbf{w} \cdot \mathbf{x} + b = 0 \tag{1}$$

を考える．
正，負の学習データをそれぞれ 2 つの領域に分割する．

$$\mathbf{w} \cdot \mathbf{x}_i + b \geq 1 \quad \text{if } y_i \in 1 \tag{2}$$

$$\mathbf{w} \cdot \mathbf{x}_i + b \leq -1 \quad \text{if } y_i \in -1 \tag{3}$$

式 (2),(3) をまとめると，

$$y_i \left(\mathbf{w} \cdot \mathbf{x}_i + b\right) \geq 1 \quad (i = 1, \cdots, l) \tag{4}$$

式 (4) に属さない領域がマージンの領域である．ここで式 (1) の分離超平面と個々の学習データ $\mathbf{x}_i$ の距離 $d(\mathbf{w}, b, \mathbf{x}_i)$ は

$$d(\mathbf{w}, b, \mathbf{x}_i) = \frac{|\mathbf{w} \cdot \mathbf{x}_i + b|}{\|\mathbf{w}\|}$$

となる．マージン幅を分離超平面と両クラスの中で最も分離超平面に近い学習データとの距離とすると

$$\min_{\mathbf{x}_i(y_i=1)} d(\mathbf{w}, b, \mathbf{x}_i) + \min_{\mathbf{x}_i(y_i=-1)} d(\mathbf{w}, b, \mathbf{x}_i) \quad (i = 1, \cdots, l)$$

$$= \frac{1}{\|\mathbf{w}\|} + \frac{1}{\|\mathbf{w}\|} = \frac{2}{\|\mathbf{w}\|}$$

マージンを最大化するには $\|\mathbf{w}\|^2$ を最小化すればよい．以上のことからこれは次の制約付き最適化問題に定式化できる．

$$\begin{aligned} &\text{minimize} && \|\mathbf{w}\|^2 \\ &\text{subject to} && y_i \left(\mathbf{w} \cdot \mathbf{x}_i + b\right) \geq 1 \quad (i = 1, \cdots, l) \end{aligned}$$

この制約付き最適化問題は Lagrange 乗数を用いると，より扱いやすい双対問題に帰着できる．
miximize

$$L(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j)$$

subject to

$$\alpha_i \geq 0, \quad \sum_{i=1}^{l} \alpha_i y_i = 0 \quad (i = 1, \cdots, l)$$

この式で $\alpha_i \geq 0$ となる $\mathbf{x}_i$ を Support Vector と呼ぶ．$\mathbf{w}$ はこれらの Support Vector より

$$\mathbf{w} = \sum_{i(ALLSV)} \alpha_i y_i \mathbf{x}_i$$

となり，b は

$$b = -\left(\mathbf{w} \cdot \mathbf{x}_i + \frac{1}{y_i}\right)$$

で求められる

2.4 SVM の拡張

2.3 節で述べた方法は全てのパターンが分離超平面で完全に分割可能であることを前提としている．しかし対象のパターンの集合が必ずしもそうであるとは限らない．SVM では分離超平面で分割しきれないパターンに対して以下の 2 通りの手法を用いて対処する．これらの手法を用いることで SVM はより一般的な分類に対して適用出来るようになる．

2.4.1 ソフトマージン

学習データに対する多少の識別誤差を許すように制約を緩める．各データに対し新たにスラック (遊び) 幅を示す変数 $\xi i \geq 0 \quad (i = 1, \cdots, l)$ を決め，式 (2),(3) の制約を以下のように緩める

$$\mathbf{w} \cdot \mathbf{x}_i + b \geq 1 - \xi i \quad \text{if } y_i \in 1$$

$$\mathbf{w} \cdot \mathbf{x}_i + b \leq -1 + \xi_i \quad \text{if } y_i \in -1$$

これにより制約付き最適化問題は

$$\begin{aligned} &\text{minimize} && \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^{l} \xi_i \\ &\text{subject to} && y_i \left(\mathbf{w} \cdot \mathbf{x}_i + b\right) \geq 1 - \xi i \\ & && \xi_i \geq 0 \quad (i = 1, \cdots, l) \end{aligned}$$

となる．ここで C は誤り判定による損失の重みを意味する．つまり，C が大きいときは判別の誤りをしないように分類を行い，C が小さいときは多少の判別の誤りは許し，できるだけ大きなマージンを取るようになる．C の最適な値は実験的に求められる．

これに対し，2.3 節と同様に Lagrange 乗数を用いる．
mixiimize

$$L(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j)$$

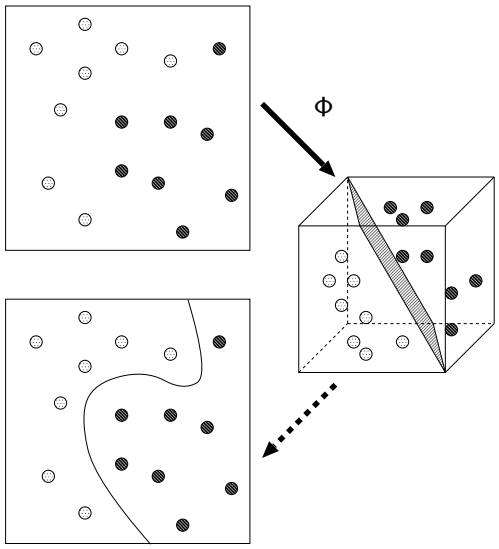
subject to

$$0 \leq \alpha_i \leq C, \quad \sum_{i=1}^{l} \alpha_i y_i = 0 \quad (i = 1, \cdots, l)$$

これによって得られた Support Vector に対して 2.3 節と同様に $\mathbf{w}$，b を求める．

2.4.2 カーネル関数

パターンをそれが属する次元よりも高次元な空間へ写像し，その空間上で分離超平面を求める．結果，元の空間上では非線形によって分割したことと同義になる．



ある入力 $\mathbf{x}$ を写像関数 Φ を用いて高次元空間に写像する．写像先の高次元空間上で $\mathbf{x}$ の位置は $\Phi(\mathbf{x})$ である．この高次元空間上で分離超平面により分割を行うので，ある未知のデータ $\mathbf{x}_u$ を識別関数にかけると

$$\begin{aligned} f(\Phi(\mathbf{x}_u)) &= \mathbf{w} \cdot \Phi(\mathbf{x}_u) + b \\ &= \sum_{i(allSV)} \alpha_i y_i \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_u) + b \end{aligned}$$

となる．
しかし考える次元が大きいと計算量は膨大になるので，実用上この計算を行うことは難しい．しかし，$\mathbf{x}_i$ と $\mathbf{x}_j$ の高次元空間への写像の内積

$$\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$$

を $\Phi(\mathbf{x}_i)$，$\Phi(\mathbf{x}_j)$ を知ること無く計算できれば計算量は大きく減る．つまり

$$\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j) = K(\mathbf{x}_i, \mathbf{x}_j) \tag{5}$$

なる関数 K を用い，

$$f(\Phi(\mathbf{x}_u)) = \sum_{i(allSV)} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}_u) + b$$

このようにして写像関数 Φ の計算を避ける手法を**カーネルトリック**と呼び，このような関数 K を**カーネル関数**と呼ぶ．関数 K が式 (5) を満たすには Mercer の定理を満たせばよいことが知られている．

[定理 Mercer]
関数 K が，ある写像関数 Φ を用いて

$$K(x, y) = \sum_{i=1}^{l} \Phi(x)\Phi(y)$$

と展開できるための必要十分条件は，$\int g(x)^2 dx$ が有限となるような任意の $g(x)$ に対して

$$\int K(x, y) g(x) g(y) dx dy \geq 0$$

が成り立てば良い．
この定理の成立の確認は容易ではないが，既に幾つかの関数が確認されている．簡単なものでは以下のような関数がある．

$$(x \cdot y + \alpha)^{\beta}$$

$$\exp(\frac{-\|x - y\|^2}{2\gamma^2})$$

$$\tanh(\kappa x \cdot y - \delta)$$

3 機械的学習の電子メールの分類への適用

2.1 節で機械的学習では特徴をそれぞれ各ベクトルにパラメータとして設定することを述べた．分類対象の特徴の何をどのようにパラメータにするかは分類器の性能を決定する重要な要素である．

電子メールはそれ自身に分類に有効な多くの情報を含んでいるが，本論文では電子メールの内容に直接関係がある『件名』と『本文』のみを特徴として扱うことにする．『件名』と『本文』を文章として考えると，文章は個々の形態素という小さな部品で構成されている．形態素とは，それ以上分割すると語句の意味が失われる語句の最小単位である．例えば次のように文章から形態素を得る．

今日は良い天気ですね．
↓
今日 / は / 良い / 天気 / です / ね / ．

日本語の文は英文とは違い各単語を空白で区切る慣習は無いため，複雑な方法を用いて形態素を得る必要がある．しかし，この問題は本論文の目的ではないので省略する．既に優れた性能をもった日本語形態素解析システムが開発されており，本論文では単にそれを使用する．代表的なシステムとしては奈良先端大 松本研究室による茶筌[1] や京都大学 長尾研究室による JUMAN[2] がある．本論文の実験では茶筌を用いた．この形態素を文章の特徴とすると，文脈によって多少の意味の違いは出るが，ある程度の文意を推定することが可能だと考えられる．

ただし，機械的学習では単純に得られた要素全てを特徴とするより，文章の分類を行う上で役に立たない要素 (ノイズ) はできるだけ除いた方が分類器の性能は高くなる．つま

[1] http://chasen.aist-nara.ac.jp/index.html.ja
[2] http://pine.kuee.kyoto-u.ac.jp/nl-resource/juman.html

りこの方法を用いた文章分類問題では全ての形態素を特徴とするより，文章の分類を行う上で役に立たない形態素 (stop word) はできるだけ除いたほうが分類器の性能が高くなるということである．それは SVM においても例外ではない [7]．茶筌では形態素の品詞も解析してくれるので品詞レベルでの stop word の設定が可能である．本論文では得られた形態素のうち，名詞 (1 文字のサ変名詞を除く)，動詞，未定義語を特徴として採用し，その他の品詞は stop word として除外した．前述の例文では『今日』『天気』の２種類の形態素が特徴として得られる．

各特徴のパラメータは，ある電子メール $\mathbf{x}_i$ 内である形態素 a_k が m 回現れた場合，そのパラメータ $P(\mathbf{x}_i : a_k)$ を

$$P(\mathbf{x}_i : a_k) = \sum_{j=1}^{m} w^{j-1}$$

とする．ただし，w は $0 \leq w \leq 1$ の範囲で設定する重複形態素に対する重みである．つまり，電子メールの特徴 $\mathbf{x}_i$ は

$$\mathbf{x}_i = \{P(\mathbf{x}_i : a_1), P(\mathbf{x}_i : a_2), \cdots, P(\mathbf{x}_i : a_n)\}$$

の n 次元のベクトルで表される．

なお，本論文では件名と本文では形態素の重要度に差があると考え，同一の形態素でも別の特徴とした．

4 実験

本節では分類対象，パラメータ等を変化させて電子メールの分類における SVM の性能評価を行う．性能評価には適合率 (Precision)，再現率 (Recall)，精度 (Accuracy) を用いる．ただし，データを正と負に分類する場合，適合率は分類器が正であると判別して実際にそうであったものの割合，再現率は正に属する全データのうち分類器が正しく判別できた割合である．

4.1 設定

4.1.1 分類対象

実験に使用するデータ用として，表 1 のような 4 種類のメーリングリストと収集した迷惑メール用意した．ただしメーリングリストでは件名につけられる通し番号等は削除してある．各データセットが扱う話題について捕捉的な説明を行うと，FreeBSD と Linux は共に UNIX 系の OS，skk は日本語入力システム，Ruby はプログラミング言語である．freebsd-users-jp と linux-users では OS の話題だけでなく，プラットフォーム全般に関する話題を扱っている．skk や Ruby は FreeBSD や Linux 上でも動作するため，無関係とは言えない．

学習の際には正のデータと負のデータを必ず同数ずつ学習させ，それを『学習 : n』の形で示すことにする．例えば学習 : 100 は正のデータを 100 通，負のデータを 100 通学習させたことを示す．また，テスト用データも正と負のデータを同数ずつ与えて評価を行う．

これ以降の実験では，各データセットは表 1 の略称で表すことにする

4.1.2 SVM の設定

SVM については既に実装がされているもの[3]から Dortmund 大学で開発された SVM^{light}[4] を選んで使用した．

SVM の設定は，ソフトマージンのパラメータ C を

$$C = (\text{avg } \mathbf{x}_i \cdot \mathbf{x}_j)^{-1}$$

とし，カーネル関数は用いない．

4.1.3 重複形態素に対する重み w の決定

3 節で述べた重複形態素にかかる重み w が分類性能にどのような影響を与えるかを調べる．w の値は分類対象の特徴空間上の位置に関係する．もしも形態素の出現有無以外に出現回数も話題の推定に役に立つのならば w は 0 以外のときが分類性能が高くなるだろうし，役に立たないのならば w が 0 のときが分類性能が高くなる．

正に *skk*，負に *FreeBSD* を用い，$w = 0$ と $w = 1.0$ に対して学習と性能の変化を計測したところ図 1 の (a-1)，(a-2) ようになった．また，学習 : 1000 に固定したときの w に対する性能の変化が図 1 の (b) である．

性能は $w = 0$ のときに最大になり，w が大きくなると性能が下がることが分かる．このことから形態素の出現回数の情報は，話題によって電子メールの分類を期待する場合には意味を持たない情報 (ノイズ) であると考えてよいだろう．

以降の実験では $w = 0$，つまり，ある形態素が出現したかどうかの情報だけを持つことにする．$w = 0$ のとき，$P(\mathbf{x}_i : a_k)$ は以下と同義である．

$$P(\mathbf{x}_i : a_k) = \begin{cases} 1 & \text{if} \quad m > 0 \\ 0 & \text{if} \quad m = 0 \end{cases}$$

4.2 データセットの分析と SVM の性能

SVM による分類は特徴空間の分割であるため，分類対象の特徴空間上での分布を知ることは分類器の性能を評価する上で必要である．

各データセットの分析を始める前に話題と分類のしやすさの関係を考えてみる．全話題 $\forall T$ を 2 つのクラスに分類する場合

$$\forall T = C_1 : T + C_2 : T$$

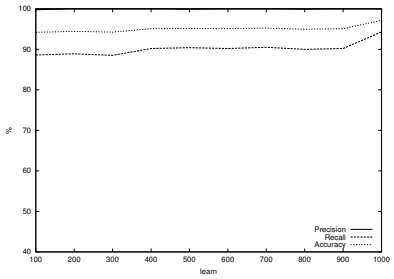
のようになる．これらが分類しやすいのは $C_1 : T$ と $C_2 : T$ が全く異なっている場合である．つまり $C_1 : T$ と $C_2 : T$ で，持つ形態素と持たない形態素がはっきり分かれている場合であると言える．これは特徴空間上で両者がはっきりと分離して分布することを意味するためイメージ的にも簡単に分類できる．しかし両者にあまり違いが無い場合，両者の分布が近付いたり，混ざりあった分布を取るため分類の性能は低下する．

今回のデータセットについて分類性能を予測してみる．*FreeBSD* と *Linux* は両者とも同じような話題であるため分類が難しく，逆に *skk* と *Ruby* はほとんど接点が無いため分類が容易だと予測できる．これを確かめるため，正と負に用いるデータセットを変え，学習に対する性能を計測した結果が図 2 である．ただし，この実験では内容による分類

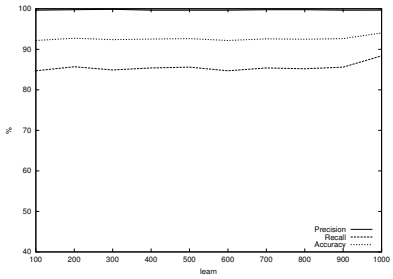
表 1: 実験用データの種類

略称	データセット	主な話題
FreeBSD	freebsd-users-jp	FreeBSD 全般の話題
Linux	linux-users	Linux 全般の話題
skk	skk ml	skk の話題
Ruby	ruby_list	Ruby の話題

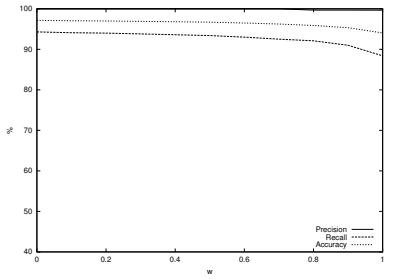
[3] http://www.kernel-machines.org/
[4] http://svmlight.joachims.org/



(a-1) $w = 0$

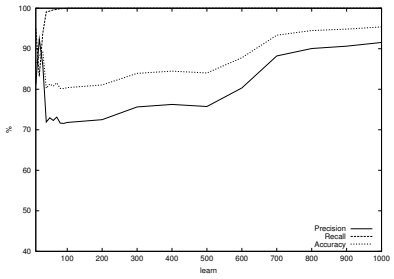


(a-2) $w = 1.0$

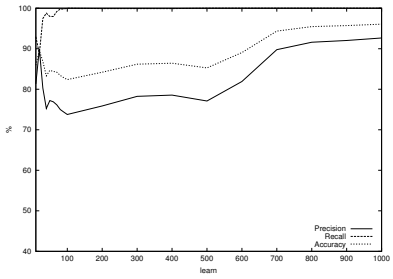


(b) 学習 : 1000 における w の影響

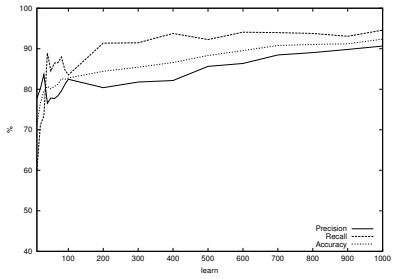
図 1: w の値による性能の変化



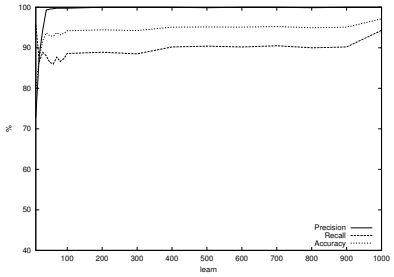
(a) *FreeBSD* / *Ruby*



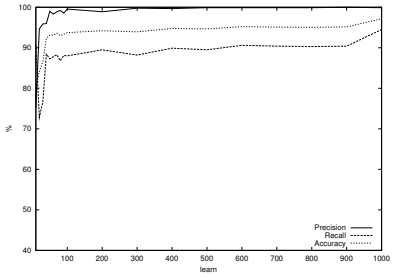
(b) *Linux* / *Ruby*



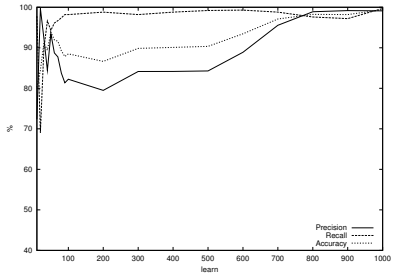
(c) *FreeBSD* / *Linux*



(d) *skk* / *FreeBSD*



(e) *skk* / *Linux*



(f) *skk* / *Ruby*

図 2: 各種データセットの比較

という点から見ると性能に多少の誤差が生じる可能性がある点は留意しておく必要がある．メーリングリストの特性から大部分は内容によって正しく分類されたと見て良いが，中には例外的な相応しくないデータが存在する場合があるためである．しかし内容がどちらに属しているかの境界は曖昧であり，絶対的なものではない．そのためこの実験では誤差は無視し，単にメールの内容と属するメーリングリストは同一とした．

まず，*FreeBSD* と *Linux* の話題が似ているということを確かめてみる．図中の (a) と (b)，(d) と (e) を見ると，*FreeBSD* と *Linux* が変わってもグラフの形にあまり変化が見られない．さらに (c) を見ると全グラフ中最も性能が良くないことが分かる．このことから *FreeBSD* と *Linux* は話題が似ていると言ってもよいであろう．

次に (f) に注目してみる．(f) の性能は始めのうちは (d) や (e) に及ばない．しかしグラフの形に注目してみると (f) の形状は (a)，(b) と似ていることが分かる．このことから，始めの方の *Ruby* 側の学習用データがあまり特徴を捉えていない良くない学習データだったと推測できる．それにも関わらず，(f) は最終的には精度が 99.40% と頭一つ抜けた性能を出しており，これも予測と合致する．

ところで，(d) と (e) では適合率がほぼ 100% であるのに対して再現率は低く，(a) と (b) ではその逆である．これは SVM が *skk*，*Ruby* と判定したものについてはほぼ間違が無く，*FreeBSD* と *Linux* と判定したものに間違いが生じていることを示す．この偏りは両クラスが持つ決定的な特徴の差異によって生じると考えられる．*Linux* や *FreeBSD* で扱う話題を計算機について広く扱った一般的な話題と捉えると，*skk* や *Ruby* で扱う話題はその中の一部を専門的に扱った話題と見ることが出来る．一般的な話題に偏って出現する形態素はその話題の広さのため分散しており，少数では話題を判別する特徴としては信頼性が低い．逆に専門的な話題に偏って出現する形態素はある程度の出現数を持ち，一般的な話題に出現することはあまり無いのでクラスを判別する特徴として信頼性が高い．図 3 は縦軸に専門的な話題に偏って出現する特殊な形態素の量，横軸に一般的な話題に偏って出現する形態素の量をとったときの (a)，(b)，(d)，(e) のような場合の分布のイメージ図である．黒が一般的な話題，白が専門的な話題を示す．これを精度良く分類する場合には破線のように特殊な形態素を基準に分割するのが良いであろう．するとこの実験のように専門的な話題のように，決定的な形態素があるものはほぼ正しく分類されるようになると考えられる．

なお，通し番号を削除しなかった場合の分類はいずれも 100% の分類性能を発揮した．

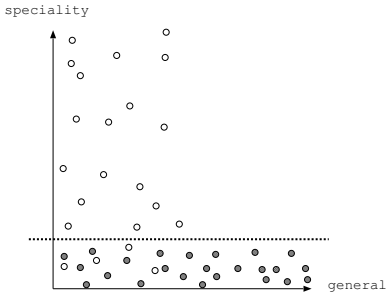


図 3: 偏った分類のイメージ

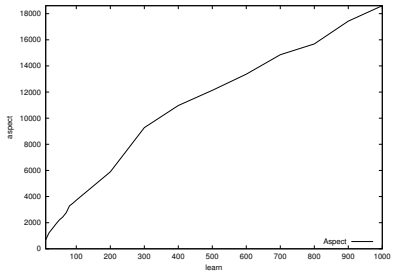
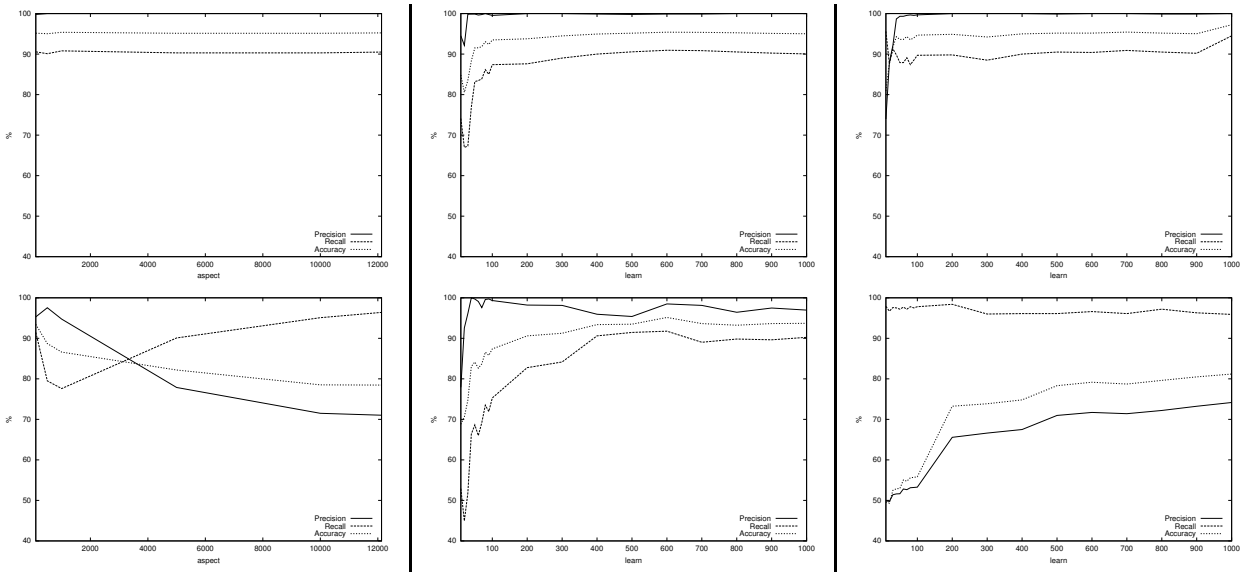


図 4: 学習と特徴数



(a) 特徴数と性能　(b) 学習と性能 (特徴を 100 に固定)　(c) 学習と性能 (全特徴)

図 5: SVM と NN 法の比較 (上:SVM，下:NN 法)

4.3 NN 法との比較

SVM の分類器としての特性を確かめるため，最も単純な機械的学習である NN 法 (Nearest Neighbor Rule) との比較を行う．比較に用いる実験用データは，正に *skk*，負に *FreeBSD* を用いた．また比較を単純にするために件名は用いず本文のみを特徴とした．本論文で用いる手法では学習を変化させると特徴数は学習と共に増加する．学習に対する特徴数は，この設定の場合図 4 のように増加した．

特徴数と分類性能の関係を調べる．l 個の学習を行った時に，ある特徴 a_k の重要度 $I(a_k)$ を

$$I(a_k) = \left| \sum_{i=0}^{l} y_i P(\mathbf{x}_i : a_k) \right|$$

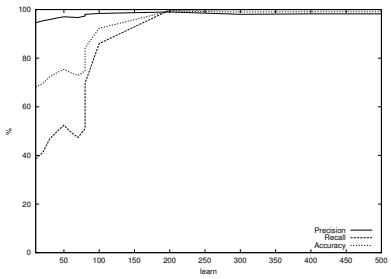
とし，この重要度 $I(a_k)$ を上位から順に取ることで特徴を絞り込む．学習 : 500 に固定し，上記の方法で有効とする特徴数を変化させて性能の変化を計測した結果，図 5 の (a) のようになった．NN 法では特徴数が増えるにつれて適合率と再現率のトレードオフが起きているのが分かる．ただし特徴数が多くなるにつれ，全体の精度は徐々に下がっていく．これは次元が増え，空間が増大するのに対して十分なサンプル数が与えられないことが原因である．それに対して SVM では特徴数によらずほぼ一定の性能となっており，SVM の汎化性能が次元数によらないことが分かる．

このことを別の方向から確かめてみる．同様の方法で特徴数を $I(a_k)$ の上位 100 に固定し，学習に対する性能の変化を計測してみると図 5 の (b) ような結果になった．この方式の場合 NN 法もまずまずの性能を発揮しているが，SVM の性能を越えることは無かった．特徴数を絞らずに学習をした場合の性能の変化を計測した結果が図 5 の (c) である．この場合 NN 法は学習効率を大幅に落しており，学習 : 1000 の段階でもその精度は 81.18% にしかなっていない．SVM では図 5 の (b) の結果とあまり変わらず，むしろ多少良い結果を出しているのが分かる．

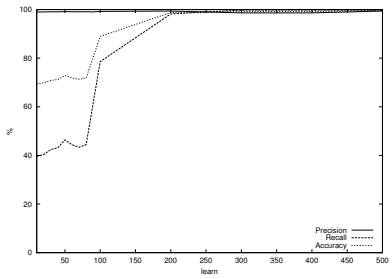
NN 法を始め多くの機械的学習では，特徴の絞り込みによって特徴空間の削減を行わなくては性能が大きく低下する．しかし SVM では次元数によらず汎化能力の高い分離超平面を求めることができる．

4.4 迷惑メールの分類に対する性能

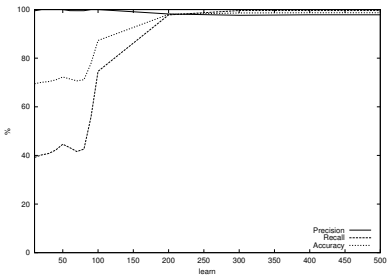
迷惑メールの分類を実際を行い，SVM の実運用上の有用性について検証を行う．分類対象として各データセットに加え，全データセットを等分に学習させたものと，それに通し番号を削除した数種類のメールマガジンを同数混ぜたものとの比較も行った．その結果を図 6 に示す．



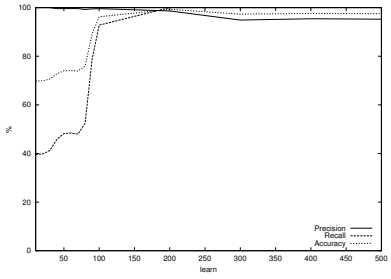
迷惑メール/skk



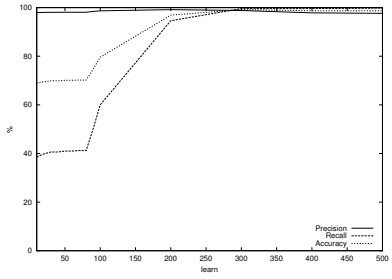
迷惑メール/$FreeBSD$



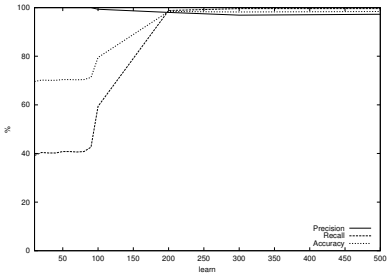
迷惑メール/$Linux$



迷惑メール/$Ruby$



迷惑メール/全データセット



迷惑メール/全データセット+メールマガジン

図 6: 迷惑メールとの分類

いずれも最終的には高い精度で分類ができている．これは本論文で用いた手法と同様に，形態素を用いた文章分類と比較して高い性能である．これには SVM の性能もさることながら，文章としては短いものが多いという電子メールの特性も作用していると考えられる．文章が短いほど形態素の情報と文意のずれは少なくなると考えられるからである．

それぞれの分類性能を比較をすると学習の効率がやや異なっていることが分かるだろう．4.2 節での結果と照らし合わせると，話題が広く一般的なものの方が学習の効率が低くなっていることが分かる．ただし話題の幅が広がったにも関わらず，メールマガジンを混ぜたものについてはこの限りでは無かった．通し番号の情報を削除しても，メールマガジンの場合は本文中にも必ず出現する決定的な要素が入っており，それによって判別されてしまったためだと考えられる．

特筆することとしてどの分類でも初期の段階から適合率が非常に高いことが挙げられる．つまり，SVM が迷惑メールであると判定したものは，そのほとんどが迷惑メールであった．迷惑メールはメーリングリストのように話題の範囲が限定されていないため，特徴が掴みにくく正確な分類がしにくいのではないかと予想していたので意外な結果であった．得られた特徴を $I(a_k)$ の値を基に分析してみると，迷惑メール側に偏った形態素が多くなることが分かった．例えば迷惑メールと，最も特徴が掴みやすいと予想される skk で学習 :500 の時，$I(a_k)$ の上位 10 のうち 9 は skk 側に偏った形態素であったが，上位 100 を見るとほぼ同数になり，上位 1000 を取ると 2/3 は迷惑メールに偏った形態素であった．$I(a_k)$ の順位や値は決定的な特徴であるかどうかを調べる一つの目安にしか過ぎないため，はっきりとしたことは言えないが，今回用いたどのデータセットよりも迷惑メールの方が強い特徴が出ているという推測ができる．よって実運用上は，よほど分類対象が迷惑メールに近いもので無い限り適合率は同じような傾向になると考えられる．迷惑メールのフィルタリングの場合，最も損失が大きいのは迷惑メールでないメールを誤って迷惑メールと判定してしまうことであると考えられるので，適合率が高くなるのは良い傾向と言える．なお，性能がほぼ飽和した後に，学習によって適合率が多少下がっているものもあるが，データセットによっては僅かに迷惑メールが含まれているため，本当に誤った分類をされているメールの総数はそれほど変わらなかった．

誤った分類が行われたメールについて検証をしてみると，大まかに 3 つのタイプに分かれることが分かった．

- 判別不能型
  本文が一言，あるいは一単語しかない．
- 類似型
  異なったクラスの成分を多く持つ
- 学習不足型
  まだ学習が不十分なため誤った分類が行われる．

判別不能型にはメーリングリストに誤って混入したコマンドメールや，発言のテストだけを行ったメール，類似型には今回の場合ソフトウェアの広告等があった．学習不足型は内容が他と比べ特異なものが多い傾向があった．

学習初期段階の誤りで大半を占めたのは学習不足型であった．学習不足型は学習を進めて行くうちに減少する傾向にあり，学習 : 500 の段階では多くても 1 通程しか無かった．学習が進んでもほとんど減らなかったものは判別不能型であるが，これは前処理で例外としておくことで十分対処が可能だと考えられる．類似型は学習が進むことで増えたり減ったりした．これは形態素と文意のずれが原因であるため，この実験のように形態素を用いたフィルタリングでは防ぐことが難しく，一番の難点であると言えるだろう．

5 システム例

4.4 節で示したように SVM は電子メールの分類にも非常に高い性能を発揮した．勿論 4.4 節で用いた分類対象は迷惑メールでないものを一般メールと定義すると，範囲は非常に限定されたものである．そのため広大な範囲を持つ一般メールと迷惑メールの分類ではこれほどの高い精度は出ないと思われる．しかし，個人が使う電子メールはある程度範囲が限定されているのが普通であり，近い分類性能は十分に期待できるだろう．また，話題が限定されたメーリングリストに迷惑メールの混入を防ぐようなシステムとしては非常に有効であると考えられる．

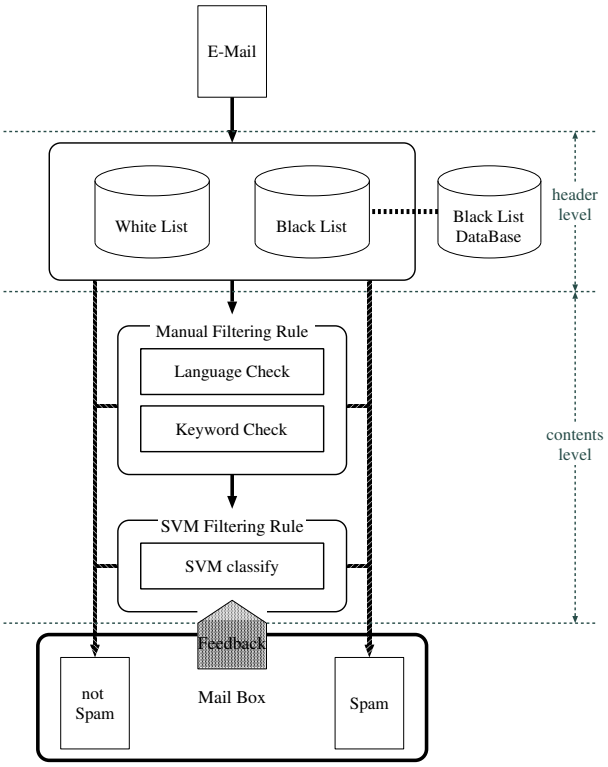


図 7: システム例

さらに 4.3 節で示したように，SVM は特徴数の削減を必要とせずに汎化性能の高い分類が可能である．そのため，追加学習の際に有効特徴の再検証と特徴空間の再構築を必要としない．これにより学習の際に計算機にかかるコストが大幅に軽減された．個人レベルのコンピュータでも気軽に追加学習によって性能の向上や調整を計ることができるのは大きなメリットである．

しかし，4.4 節において，迷惑メールと全データセットとの分類性能が飽和するまでには 学習 : 200 以上が必要であった．学習データの質や分類対象にもよるが，今回用いたデータセットよりも複雑なものと分類する場合にはさらに多くの学習が必要であろう．一般メールはともかく，個人が十分な数の迷惑メールを集めて学習させるのは難しい．機械的学習によるフィルタリングを実用化するには，予め迷惑メールを収集したデータベースが必要になるだろう．

具体的にこの手法を用いたフィルタリングシステムの例を図 7 に示す．SVM を用いたフィルタリングには学習データが必要になるので，単体で用いるには学習の初期段階では不安がある．そこで既存のフィルタリングシステムと組合せることでお互いの弱点を補うようなものが考えられる．既存のフィルタリング手法は学習初期段階の分類精度を補うだけでなく，学習の半自動化にも役に立つ．また，逆に SVM による分類の結果を基にブラックリストを自動的に更新する等も考えられる．それだけでなく，両方を用いることによってさらに高い精度の分類も可能になるかもしれない．例えば本実験では実験対象に日本語と英語を混在させたが，Chasen では英語は全て未定義語として取得されるため，英文に対しては特に stop word の設定は行っていない．予め文字コード情報を基にふるい分けを行い言語ごとに適した特徴抽出と学習を行えば，さらに精度を上げることが可能であると考えられる．

6 おわりに

SVM は電子メールのフィルタリングにも高い性能を発揮することが分かった．学習の問題があるため万能とは言えないが，既存のフィルタリングを組み合わせる等の工夫を行えば十分実用に値するものになると筆者は考える．

本実験でも SVM のソフトマージンのパラメータ C の調整やカーネル関数の使用による性能の向上を試みたが，残念ながらこれらの設定と精度の関係を発見するには至らなかった．これは今後の課題と言えるだろう．

参考文献

[1] 総務省 : 特定電子メールの送信の適正化等に関する法律

[2] 経済産業省 : 特定商取引に関する法律 (改正)

[3] 石井 健一郎, 上田 修功, 前田 英作, 村瀬 洋 : わかりやすいパターン認識, オーム社, SBN 4-274-13149-1

[4] Cortes,C. and Vapnik,V. : Support Vector Networks, *Machine Learning*, Vol.20, pp.273-297, 1995

[5] Vapnik,V. : The Nature of Statistical Learning Theory, Springer-Verlag, 1995

[6] 平 博順, 向内隆文, 春野雅彦 : Support Vector Machine によるテキスト分類, 情報処理学会論文誌, NL128-24, pp.173-180(1998)

[7] 平 博順, 春野 雅彦 : Support Vector Machine によるテキスト分類における属性選択, 情報処理学会論文誌, ISSN 0387-5806, pp.1113-1123(2000)

[8] Open Relay Database : http://www.ordb.org/