Vol.2013-ICS-171 No.2
2013/3/18

複数単語共起フィルタリングにより大規模化するデータを処理する有害文分類手法の提案

Deyue Deng[1,a)]　大塚 孝信[1,3,b)]　伊藤 孝行[1,2,3,c)]

概要：近年行われた情報フィルタリングの研究では，共起情報を導入し，フィルタリングの性能を上げるには，学習データが大規模となることで，膨大となる処理時間への対応が課題となっている．したがって，共起の単語数を増加させた場合，爆発的なデータ数となる共起を高速に処理する方法を考えることが必要である．ストレージの容量的問題もあるが，データの処理，プログラムとデータベース間協調の効率を向上することが重要となる．また，共起の増加に伴いノイズも増加するので，処理性能に支障をきたす可能性もある．本研究では既存手法では構築困難な大規模な共起データを本扱う手法を提案する．提案手法では，並列処理を用いて，実用的範囲で許容できる時間内学習過程や判定過程を実行できるフィルタリングを実装できる．

1. はじめに

計算機による有害文書の自動判定は，文書の判定精度が低く，文法的に曖昧な文書処理や，新たに出現する有害な単語への，迅速な対応が難しい．既存の文書分類手法の多くは，曖昧な文書や新たに出現した単語に対応しきれないという大きな問題点を抱えている．そこで，本研究は，以上の問題を解決するためにベイジアンフィルタ上で共起情報を用いる手法 [1] を導入した．共起とは，文書の中に複数の単語が同時に出現することである．1 つの単語の出現確率だけ統計するベイジアンフィルタでは，隠語や造語などで構成される文書を判断することが難しい．しかし共起では複数単語の出現確率を統計するため，文脈を推測することができ，精度の向上が期待できる．**図 1** に共起の例を示す．「スピード」という単語が複数の意味で用いられている，負例の例で，有害な意味合い (麻薬) で使われている．本提案手法では共起を用い他の単語と組み合わせることで負例として判別できている．共起を用いることで，1 単語の出現確率を用いるだけで判断できない文書も容易に判断できる．

フィルタリングの性能を向上する場合，大規模学習データの処理時間が膨大となることが大きな課題である．共起

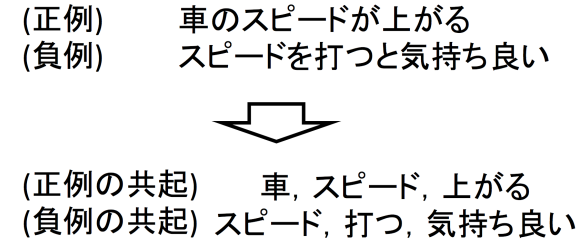

図 1　共起の例

の単語数を増加させた場合，膨大なデータを高速に処理する方法を考える必要がある．例えば，2 万件の学習データを取り扱う場合，単純に計算しても，単語の総数は 1,280,278 個となり，2 単語共起の数は 1.14 億である．3 単語共起では，共起数は 134 億となる．ストレージの容量的問題もあるが，データの処理，プログラムとデータベース間の協調を効率化することが重要となる．さらに，共起の増加に伴いノイズも増加するので，処理性能に支障をきたす可能性も大きい．

本研究ではまず，既存手法 [1] では構築が困難な大規模の共起データを提案手法を用いて，データベースへ格納する．その上で，並列処理などの手法を用いて，実用的範囲での時間内学習過程や判定過程を実行できるフィルタリングを実装する．

評価実験では，既存手法のベイジアンフィルタリング [3][5]，2 単語共起フィルタリング (2 万件学習データ) [1]，及び提案手法で構築した大規模データベース (20 万件

[1] 名古屋工業大学大学院産業戦略工学専攻
[2] 東京大学政策ビジョン研究センター
[3] 名古屋工業大学 グリーン・コンピューティング研究所
[a)] deng@itolab.nitech.ac.jp
[b)] otsuka.takanobu@nitech.ac.jp
[c)] ito.takayuki@nitech.ac.jp

学習データ) を用い，2 単語共起と 3 単語共起フィルタリングを比較する．以上によって，精度とデータベースの構築時間の短縮を両立させた有害文書分類手法を実現する．

2. 関連研究

本研究では，教師あり機械学習 [2] を用いてフィルタリングを行う．機械学習とは，人間が自然に行っている学習能力と同様の機能をコンピュータで実現しようとする手法のことである．データベースに大量のサンプルデータ集合を入力して解析を行い，有用な規則などを抽出し，アルゴリズムを発展させる．データは統計学的手法 (ベイジアンフィルタなど) で解析する．機械学習の目的は，サンプルデータから学習した「既知」のデータの特徴に基づき「未知」のデータから同様の特徴を発見することである．

2.1 単純ベイズ分類器（Naive Bayes classifier）

単純ベイズ分類器 [3] は，強い（単純な）独立性仮定と共にベイズの定理を適用することに基づいた単純な確率的分類器である．独立性仮定の基となる確率モデルは，独立特徴モデル（independent feature model）である．単純ベイズ分類器では，上記の教師あり学習の設定で効率的に訓練可能である．一般に有害文書分類で用いる，単純ベイズ分類器の原理について述べる．まず，文書 d がカテゴリ C に属する確率を $P(C|d)$ とし，$P(C|d)$ をベイズ定理を適用し，計算する (式 1).

$$P\left(C|d\right) = \frac{P(C) \times P\left(d|C\right)}{P(d)} \tag{1}$$

周辺確率 $P(d)$ は特定の文書にとって一定 (変化しない) のため，カテゴリに依存せず，必ずしも計算することはない．

$P(d|C) \times p(C)$ は以下のように表され，同時確率の式 (2) と等価である．

$$P(C, d) \tag{2}$$

文書 d を単語の集合としてモデル化するなら，

$$d = (w_1, w_2, w_3, ..., w_n) \tag{3}$$

式 (3) を条件付き確率の繰り返しするとなり，式 (4) のように書き変えられる．

$$\begin{aligned} & P(C, d) \\ =\ & P(C, w_1, w_2, w_3, w_4, ..., w_n) \\ =\ & P(C)P(w_1, w_2, w_3, w_4, ..., w_n|C) \\ =\ & P(C)P(w_1|C)P(w_2, w_3, w_4, ..., w_n|C, w_1) \\ =\ & P(C)P(w_1|C)P(w_2|C, w_1) \\ & P(w_3, w_4, ..., w_n|C, w_1, w_2) \\ =\ & P(C)P(w_1|C)P(w_2|C, w_1)P(w_3|C, w_1, w_2) \\ & P(w_4, ...w_n|C, w_1, w_2, w_3) \end{aligned} \tag{4}$$

$P(C)$ は学習データの各カテゴリの文書数が総文書数に占める割合である．

文書は，単語の集合としてモデル化できる 2 つのカテゴリ正例 (無害文書) と負例 (有害文書) から取り出されるものとする．ここで文書の i 番目の単語 w_i が，カテゴリ C から取り出された文書に出現する確率は，式 (5) のように書き表せる．

$$P(w_i|C) \tag{5}$$

ここで各特徴変数すなわち単語 w_i が条件付きで他の単語 w_j $(i \neq j)$ と独立であることを仮定する．すなわち，式 (6) が成り立つとする．

$$P(w_i|C, w_j) = P(w_i|C) \tag{6}$$

したがって，式 (4) は式 (7) のように展開できる．n が単語 w_i の数である．

$$\begin{aligned} & P(C, w_1, w_2, ..., w_n) \\ =\ & P(C)P(w_1|C)P(w_2|C)...P(w_n|C) \\ =\ & P(C)\prod_{i=1}^{n} P(w_i|C) \end{aligned} \tag{7}$$

上記のような独立性の仮定の基で, 文書 d がカテゴリ C に属する確率 $P(C|d)$ は式 (8) で表わすことができる．n が単語 w_i の数である．

$$\begin{aligned} P\left(C_1|d\right) &= \frac{P(C_1) \times P\left(d|C_1\right)}{P(d)} \\ &= \frac{P(C_1) \times \prod_{i=1}^{n} P(w_i|C_1)}{\sum_{j=1}^{2} P(C_j) \times \prod_{i=1}^{n} P(w_i|C_j)} \end{aligned} \tag{8}$$

式 (8) では，問題をより簡単にするため，単語は文書中にランダムに分布すると仮定している．すなわち，単語の出現確率は，文書の長さ，文書中での他の単語との位置関係，およびその他の文脈には依存しないものとする．

2.2 Paul Graham 方式ベイジアンフィルタ

有害文書の分類は，スパムメールの分類と類似しており，大量のサンプルデータが存在する．スパムメールの検出手法として，Paul Graham によって提案された有効な誤検出を防ぐ手法 [5] である．ここではスパルメールに関する特有な部分を除き，文書分類に関するアルゴリズムについてのみ示す．まず，文書 d を単語の集合として式 (9) にモデ

ル化して示す．

$$d = (w_1, w_2, w_3, ..., w_n) \tag{9}$$

サンプルデータ中の文書で，負例で多く出現する単語と正例で多く出現する単語を標示するため，文書内に出現する各単語 w_i に対する，単語のスパム確率 $P(w_i)$ を以下の式 (10) で示す．

$$p(w_i) = \frac{\frac{B_i}{N_{Bad}}}{2 \times \frac{G_i}{N_{Good}} + \frac{B_i}{N_{Bad}}} \tag{10}$$

ただし，B_i は単語 w_i がスパムで出現した回数，G_i は単語 w_i が非スパムで出現した回数，n_{good} は学習した非スパムの総数，および n_{bad} は学習したスパムの総数である．

式 (10) での“2”は誤検出を避けるためのバイアスであり，経験則的に決められている．G_i および B_i のどちらかの出現回数が 0 であった場合は $P(w_i)$ を 0.99 および 0.01 としている．理由は，次に示す結合確率を求める際，0 あるいは 1 の場合，結合確率が 0 となり，計算できなくなることを防ぐためである．次に，求めた各単語の結合確率から最も特徴的な 15 単語を抽出し，文書 d の有害である結合確率 $P(C_{bad}|d)$ を式 (11) で求める．

$$p(C_{bad}|d) = \frac{\prod_{i=1}^{15} p(w_i)}{\prod_{i=1}^{15} p(w_i) + \prod_{i=1}^{15} 1 - p(w_i)} \tag{11}$$

単純ベイジアンフィルタと同様に，式 (11) が成立する条件は各単語の出現確率が独立であることである．

2.3 Paul Graham 方式 2 単語共起フィルタリング

安藤 [1] は，複数単語の共起を用いた判定手法を提案している．安藤の提案では，ベイジアンフィルタリングを単語のみの判定に用いているが，これまで隠語で構成される文書は，有害単語を発見することが難しいという点に着目したからであると考えられる．文章は主語，述語，目的語などの複数の単語の組み合わせから構成されており，複数の単語共起関係を判定に用いることで，文章の意味合いを判定に含めることができるとの考えに基づいている．

文章を単語だけではなく，意味合いまでを解析に含める手法として構文解析を用いる手法が提案されているが，自由に文章を書き込める場合には，文章校正などが行われないため，正しくない構文が多く出現する可能性が高く応用が難しい．例えば，ブログやコミュニティなどでは多く見られる現象である．また，ブログやコミュニティでの文書は，1 つのことを表すために正しくない構文を含めた多くの表現が含まれていることも多く，単語を ID 順にソートをし，単語の出現順序を考慮に入れることができない．例えば倒置法等が挙げられ，「猫が日向ぼっこをしている，屋根の上で」と「猫が屋根の上で日向ぼっこをしている」は，順序が違うが意味合いは同じである．上記の文書が一般的ある程度存在すると考えれば，単語の出現順序は考慮に入られない．

以上の理由から，Paul Graham 方式に共起という概念を導入した結果僅かなの性能向上が得られた．

3. 共起データベースについて

先行研究 [1] から，共起を用いたときに，計算量の増加が原因で無視できないことがかっている．共起データベースを構築する場合，共起（単語の組み合わせ）が膨大になることによって計算量が増加する．例えば，同一文書 S に含まれる単語数が n であれば，内 2 個を選ぶ組み合わせ（2 単語共起）は n(n-1)/2 通りあり，内 3 個を選ぶ組み合わせ（3 単語共起）は n(n-1)(n-2)/6 通りある．

既存の手法 [1] では膨大な数の共起データベースを構築すると，構築に大量の時間要するため実用性が乏しくなる．したがって，本研究では短時間でデータベースを構築する手法を提案する．また，構築するデータベースの処理速度を向上するため，データベースエンジンに MyISAM を用いた．さらに，計算機 5 台で並列処理することにより，共起の収集やデータベースへのアクセス処理を分散した．既存手法 [1] より 10 倍の速度に向上した．既存手法 [1] で正例と負例各 1 万件の学習データを基づいた 3 単語共起データベースを構築する場合には，20 日間以上をかかるが本提案手法を用いた場合は，同様なデータベースを 2 日間で構築することができる．

3.1 MySQL

本研究では，学習元となる正例，負例，実験に用いるテストデータ，単語を ID に紐づけるハッシュ，およびベイジアンフィルタリングのデータベースとして，MySQL でを用いた．

MySQL（マイエスキューエル）[6] は，RDBMS（リレーショナルデータベースを管理、運用するためのシステム）であり，世界でもっとも普及しているオープンソース・データベースとして知られている．日本ではこれまで RDBMS は PostgreSQL が主流であったが， 2009 年には MySQL が PostgreSQL の利用率を超えている．MySQL は，Windows，Unix，Linux，Macintosh など多くのプラットフォームで動作することが特徴の 1 つであり，また高速性と堅牢性に定評がある．Web との連携を主眼に開発されているため，Perl，PHP，Python，Ruby，Java など，多くのプログラミング言語で利用することができる．

3.2 データベースエンジン

データベースエンジン，すなわち“ストレージエンジン”はデータベース管理システム (DBMS) がデータベースに対しデータを挿入，抽出，更新および削除，するために使用する基礎となるソフトウェアである．MySQL で利用可能なストレージエンジンは数多く提案されているが，以下で

本研究で検討した 2 つのストレージエンジンを示す．この 2 つのストレージエンジンは全く異なる特長を持っている．

3.2.1 InnoDB

InnoDB(イノデービー)[7] は，MySQL 専用のデータベースエンジンであり，特徴としてトランザクション (transaction) のサポートが挙げられる．トランザクションとは，分割が不可能な情報処理の単位である．トランザクション内では，ユーザインタフェース，アプリケーションプログラム，永続性記憶資源，及び各種 I/O が実行される．トランザクション処理は，既知の一貫した状態のデータベースを維持するよう設計されており，相互依存のある複数の操作が全て完了するか，もしくは全てキャンセルされることが保証される．

例えば，顧客の普通預金口座から当座預金口座に 100 ドルを移動させる典型的な銀行のトランザクションを考えて見る．上記のトランザクションはユーザから見れば 1 つの操作であるが，コンピュータから見れば少なくとも 2 つの処理から構成されている．第一の操作は，普通預金口座から 100 ドルを引き落とす処理であり，第二の操作は当座預金口座に 100 ドルを入金する処理である．引き落とし処理が成功し， 入金処理が失敗した場合（あるいは逆の場合），銀行の帳簿（データベース）はその日の営業完了時点で不整合が発生する．したがって，トランザクション処理では 2 つの処理が両方成功するか，もしくは両方失敗することを保証する必要があり，この保証によって銀行のデータベースに不整合が生じないようにする．

3.2.2 MyISAM

MyISAM[8] では ISAM（Indexed Sequential Access Method，索引付き順次アクセス方式）に基づいているのデータベースエンジンである．高速にアクセスが可能なデータの格納方法 (ファイル編成法) の一つであり，MyISAM の元となる ISAM が IBM で開発され，今日では RDBMS に限らず， 殆ど全てのデータベース管理システム (DBMS) でデータの格納に用いられている．MyISAM を用いたシステムでは，データは固定長のレコードとして格納される．テーブルの内容へのポインタを格納したハッシュテーブルをもう一つのデータとし， 索引として用いることで，全データを検索することなく目的のデータを取り出すことを可能にしている．MyISAM はファイルへの直接の，順番に従ったアクセス方式であり，非常にシンプルであり実装も容易であり， 且つ非常に高速なアクセス方式である．しかし，それぞれのクライアントマシンがアクセスしているファイルに対して自身の接続状態を管理しなければならないという欠点を持っている．上記の欠点によって複数のデータの追加動作が衝突し，データが矛盾した状態に陥る可能性がある．したがって， MyISAM ではクライアントサーバモデルを導入することが一般的であり， これによってサーバーがクライアントの要求を直列化して扱い，問題を解決する．

本研究で使用している学習データベースを作成する場合ではトランザクション処理が必要ではないため，処理速度がもっと早い MyISAM を使用した．

3.3 形態素解析

ベイジアンフィルタの解析は単語単位でその数値を計算するため，フィルタでデータを解析する前に対象となる文章を単語単位に分解しておく必要がある．英語を含む多くの言語では，単語は普通空白によってあらかじめ“分ち書き”されるため，単語単位に分解し直す必要はない．しかし，日本語のように単語を区切るための印がない言語の場合，そのままフィルタで解析を行っても適切な解析結果が得られない. したがって，日本語の文章を適切な形に分解してからベイジアンフィルタで解析する必要がある．言語の種類によっては単語単位に分解する方法が異なるため，分解する方法の違いによってフィルタ精度に差異が発生する．上記のような課題への対応の 1 つとしてベイジアンフィルタに形態素解析（Morphological Analysis）などの自然言語処理を追加する方法がある．

形態素解析とは，コンピュータ等の計算機を用いた自然言語処理の基礎技術の 1 つであり，仮名漢字変換等にも応用されている．文法の知識（文法のルールの集まり）や辞書（品詞等の情報付きの単語リスト）を情報源として用い，自然言語で書かれた文を形態素（Morpheme， おおまかにいえば，言語で意味を持つ最小単位）の列に分割し，それぞれの品詞を判別する作業を指す．

本研究では MeCab[9] という文を形態素解析ツールを使用し，**図 2** に「すももももももものうち」を形態素解析した例を示す．

```
% mecab
すももももももものうち
すもも    名詞,一般,*,*,*,*,すもも,スモモ,スモモ
も        助詞,係助詞,*,*,*,*,も,モ,モ
もも      名詞,一般,*,*,*,*,もも,モモ,モモ
も        助詞,係助詞,*,*,*,*,も,モ,モ
もも      名詞,一般,*,*,*,*,もも,モモ,モモ
の        助詞,連体化,*,*,*,*,の,ノ,ノ
うち      名詞,非自立,副詞可能,*,*,*,うち,ウチ,ウチ
EOS
```

図 2 MeCab の出力

日本語を形態素解析する際の単語の境界判別の問題，品詞判別の問題，未知語の問題，ルーズな文法の問題などの多くの課題については，本研究の対象外とする．本研究では形態素解析ツール MeCab を利用した．また，形態素解析の際，副詞など文章を整える役割の形態素は単独では意味をなさないため，扱う単語から取り除くこととした．取り扱いの対象外とする形態素を以下に示す．

(1) 助詞
(2) 助動詞

(3) 副詞
(4) 代名詞
(5) 接続詞
(6) 数
(7) 記号

3.4 共起の定義

本研究における共起の定義を示す．共起とは，1 つの文書に出現する全ての単語の組み合わせを指す．「,」や「。」などの句読点までの 1 つの文章を共起の範囲とする定義も考慮が必要であるが，本研究は，話し言葉を分類の対象としているため，上記の定義で研究を行う．

例：「今日の天気は晴れです」という文書は，以下のような 3 単語共起になる．

- 今日, 天気, 晴れ
- 今日, 天気, です
- 今日, 晴れ, です
- 天気, 晴れ, です

安藤 [1] の先行研究により，計算量の増加による負荷が無視できないことが明らかであるが，本研究は共起する単語数を増やして，3 単語共起までの共起データベースを構築した．また，2 単語共起でも，学習データの規模を大きくし，以前より 10 倍規模のデータベースを構築した．

3 単語共起を構築する上で，最も大きな課題は共起（単語の組み合わせ）が爆発的に増加することによる計算量増加問題がある．例えば，同一文書 S に含まれる単語数が n であれば，内 2 個を選ぶ組み合わせ（2 単語共起）は $n(n-1)/2$ 通りあり，内 3 個を選ぶ組み合わせ（3 単語共起）は $n(n-1)(n-2)/6$ 通りある．

上記の方法を用いた場合，学習データとして実際 WEB のコミュニティから収集した文書 2 万件（正例，負例各 1 万件）に含まれる単語の数量は 1,280,278（内正例 552,275, 負例 728,003）となる．試算した単語共起の数を以下の**表 1** と**表 2** に示す．

表 1 10,000 件正例文書内の共起の数

単語数	552275
平均単語数	55.2275
2 単語共起の数	51,903,071（約 5,190 万）
3 単語共起の数	6,872,434,079（約 69 億）
4 単語共起の数	9.46934×10^{11}（約 9,469 億）
5 単語共起の数	1.27337×10^{14}（約 127 兆）

単語数は文書の平均単語数 10,000 の数値になる．共起の数は各文書内の平均単語数より各文書内の共起数を算出され，10,000 倍にした数値である．

既存の手法を用いて上記の共起データベースを構築すると，膨大な数の共起を処理するために大量の時間を要する．

表 2 10,000 件負例文書内の共起の数

単語数	728003
平均単語数	72.8003
2 単語共起の数	62,463,921（約 6,246 万）
3 単語共起の数	6,495,882,894（約 65 億）
4 単語共起の数	7.1399×10^{11}（約 7,140 億）
5 単語共起の数	7.8315×10^{13}（約 78.3 兆）

したがって，本研究では新たなデータベースの構築手法を提案し，3 単語共起までのデータベースを構築することとした．

3.5 共起データベースの構築手法

共起データベースの構築は以下のステップ 1，ステップ 2，ステップ 3，ステップ 4，およびステップ 5 の流れで作った (**図 3**)．

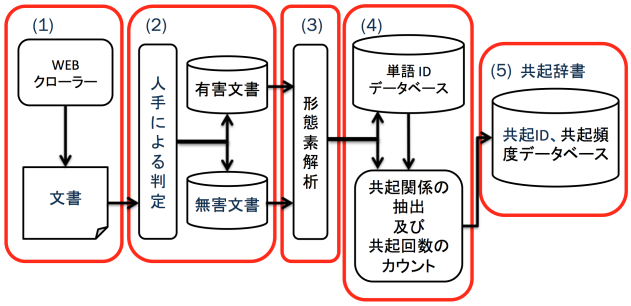

図 3 共起データベース構築の流れ

ステップ 1

学習データは，Web 上からクローラーを用いて学習データを自動で収集し，収集に用いるサイトは 2 ちゃんねる掲示板とした．今回の実験では約 22 万件の学習データを収集した．

ステップ 2

収集した学習データを正例（無害文書），負例（有害文書）に分類する．分類は，まず自動的に有害文書を分類し，次に分類した文書以外の文書は目視で分類した．学習データを収集する段階で行う自動分類は，ストップワード方式（自然言語を処理するにあたって一般的であるなどの理由で処理対象外とする単語）を用い，特定の単語が文書に出現する場合は，該当する文書を有害文書と見なすこととした．今回ストップーワードは約 200 種類収集した．目視では，有害文書と見なされなかった文書に対して，有害，無害，及び分類不明の 3 つの項目の分類を，20 人程度で作業した．

ステップ 3

形態素解析器を用いて学習データを単語分割する．本研究では，MeCab を形態素解析器に用い，’助詞’ 及び’助動詞’ と解析された単語を文書から取り除いた．

ステップ 4

単語及び共起関係のハッシュ化では，分割した単語のハッシュ化を行う．ハッシュ化を行った後，学習データの単語を全て数字に置き換える．さらに，学習データの 1 文書内にある全ての単語の組み合わせを共起関係として，ハッシュ化する．

ステップ 5

共起回数のカウントでは，共起関係をカウントし，データベースに格納する．

本研究が用いた単語ハッシュと共起データベースの構造を以下の**表 3** と**表 4** に示す．

表 3 ID ハッシュ後の 2 単語共起データベースの構造

種類	データ型
単語 ID1	INT
単語 ID2	INT
正例での出現回数	INT
負例での出現回数	INT

単語 ID1，単語 ID2 は 2 単語共起を組成される 2 つの単語を ID ハッシュした ID である．出現回数は ID1 と ID2 が組成した共起が正例，負例での出現回数である．4 つのデータはすべて INT(整数) で表れる．

表 4 ID ハッシュ後の 3 単語共起データベースの構造

種類	データ型
単語 ID1	INT
単語 ID2	INT
単語 ID3	INT
正例での出現回数	INT
負例での出現回数	INT

単語 ID1，単語 ID2，および単語 ID3 は 3 単語共起を組成される 3 つの単語を ID ハッシュした ID である．出現回数は ID1，ID2，および ID3 が組成した共起が正例，負例での出現回数である．6 つのデータはすべて INT(整数) で表れる．

3.6 提案手法と既存手法でデータベース構築速度の比較

MySQL 用の 2 つデータベースエンジン InnoDB と MyISAM を比較する．共起データベース構築の速度を向上するため，**表 5** の 1 台をサーバーと s， **表 6** の 4 台をクライアントとした．合計 5 台の計算機を利用した．

テスト用のデータは，収集した学習データから無作為に抽出した文書 10 件であり，その 10 件の文書の中に含まれていた 3 単語共起 6,239,965 個である．計算機を複数台使用する際に，全ての計算機は同じデータを入力データとしてテストを実行する．各手法の所要時間は，Java を用いた

表 5 計算機環境 1

CPU	Intel Core i7 930
Memory	24GB
HDD	7200rpm 500G
OS	Ubuntu

表 6 計算機環境 2

CPU	Intel Core 2 Duo
Memory	8GB
HDD	7200rpm 500G
OS	Mac OSX

プログラムでテストデータを入力してから，データベースの構築が完了するまでの所要時間を測定した．各手法の速度は，上記の共起数を各手法による所要時間で割った値である．チューニングのパラメータとして，SQL クエリデータ 1 回毎の Insert 数を設定する． 1 回に最大 5 万件ずつ，1 回に最大 10 万件ずつ，および 1 回に最大 100 万件ずつの 3 つの設定を比較した．テストした結果を**図 4** に示す．

グラフの青いバーは InnoDB であり，赤いバーは MyISAM である．横軸はデータベースへのアクセス速度である．縦軸は計算機 1 台，計算機 2 台，計算機 3 台，および計算機 5 台でのデータベースへのアクセス速度である．本研究が提案する手法 (計算機 5 台) は，既存手法 (計算機 1 台の青いバー) と比べて，データベースへのアクセス速度は 10 倍ほど向上した．

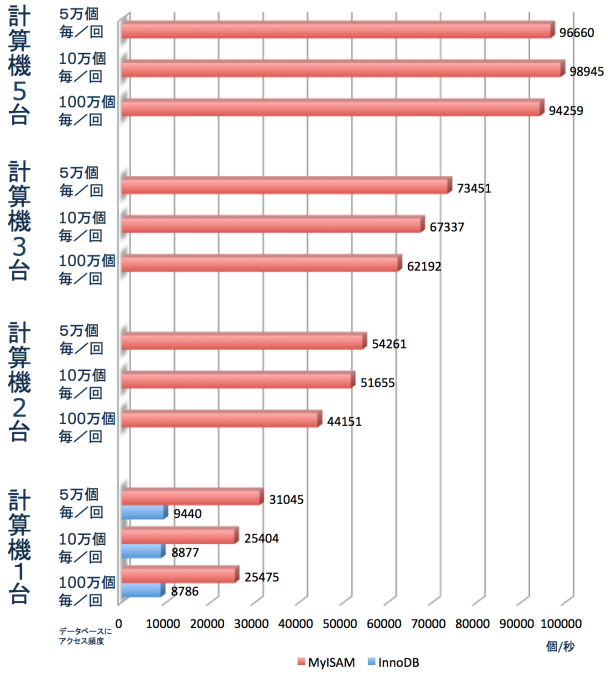

図 4 各データベースエンジンの比較

4. 複数単語共起フィルタリングの実装

まず本研究での有害文書フィルタリングで用いる共起の有害度の求め方について述べる．提案している複数単語共起を用いたフィルタリングの有害度は，ベイジアンフィルタリングを元に，複数単語間の共起を導入し，有害度の計算を行っている．

まず，2 共起を用いた場合の，各共起の有害度を求める式を式 (12) に示す．ただし，$N_p(w_i, w_j)$ は単語 w_i と単語 w_j の共起が正例で出現した回数，$N_n(w_i, w_j)$ は単語 w_i と単語 w_j の共起が負例で出現した回数，N_p は正例の総数，N_n は負例の総数である．

$$P(w_i, w_j) = \frac{\frac{N_n(w_i,w_j)}{N_n}}{\frac{N_p(w_i,w_j)}{N_p} + \frac{N_n(w_i,w_j)}{N_n}} \quad (12)$$

本研究で提案する 3 共起を用いた場合の各共起の有害度を求める式を式 (13) に示すただし，$N_p(w_i, w_j, w_k)$ は単語 w_i, 単語 w_j, および単語 w_k の共起が正例で出現した回数，$N_n(w_i, w_j, w_k)$ は単語 w_i, 単語 w_j, および単語 w_k の共起が負例で出現した回数である．

$$P(w_i, w_j, w_k) = \frac{\frac{N_n(w_i,w_j,w_k)}{N_n}}{\frac{N_p(w_i,w_j,w_k)}{N_p} + \frac{N_n(w_i,w_j,w_k)}{N_n}} \quad (13)$$

式 (12) および式 (13) では，正例あるいは負例の出現回数が 0 回の場合，式の値が 0 とならないように式 (14) および式 (15) を用いる．具体的に，ラプラススムージングという手法を用い，正例及び負例での出現回数の加算を行った．ラプラススムージングでは，それぞれの出現回数を+1することで 0 になることを防いでいる．同手法を用いた式 (14) と式 (15) に示す．

$$P(w_i, w_j) = \frac{\frac{N_n(w_i,w_j)+1)}{N_n}}{\frac{N_p(w_i,w_j)+1}{N_p} + \frac{N_n(w_i,w_j)+1}{N_n}} \quad (14)$$

$$P(w_i, w_j, w_k) = \frac{\frac{N_n(w_i,w_j,w_k)+1}{N_n}}{\frac{N_p(w_i,w_j,w_k)+1}{N_p} + \frac{N_n(w_i,w_j,w_k)+1}{N_n}} \quad (15)$$

式 (14) と式 (15) で計算した共起の有害度を用い，Naive Bayes 方式の計算式（式 (8)）と Paul Graham 方式の計算式（式 (11)）で文書毎有害度を計算する．

既存手法の Naive Bayes 方式で有害度を計算する場合，式 (8) を用いて有害度を計算する際，式 (14) と式 (15) で計算した共起の有害度 $P(w_i, w_j)$，$1 - P(w_i, w_j)$，$P(w_i, w_j, w_k)$ および $1 - P(w_i, w_j, w_k)$ を乗算する必要がある．式 (8) の n が単語 (共起) の数量である．n が一定量を超えると，共起の有害度の乗算を計算する時，倍精度浮動小数点数 (Double 型) 取り扱える範囲を超えてしまうため，0 に近づいていく．2 単語共起フィルタと 3 単語共起フィルタでは Naive Bayes 方式を使えなかった．

倍精度浮動小数点数を用いることで生じる誤差を無くすために，本研究では誤差を完全にコントロールできる BigDecimal[4] という Java のクラスを導入した．BigDecimal は任意の長さの 10 進数の表現し操作するための Java のクラスである．BigDecimal は，算術，スケール操作，丸め，比較，ハッシング，および書式変換の演算を提供している．すべての算術演算子では，演算は，まず正確な中間結果を計算し，次に選択された丸めモードを使用して (必要な場合は) 精度設定で指定された桁数に丸めるという手順で実行される．計算の途中ユーザーが丸め動作を完全に制御できるようにし，誤差なしでの計算することが可能である．BigDecimal の導入により，2 単語共起フィルタ，3 単語共起フィルタでも Naive Bayes 方式が扱えるようになった．

5. 評価実験

5.1 評価指標

本研究では，文書分類手法の評価指標として，テストデータから有害文書を分類する際の評価結果の指標 F 値を用いる．F 値は適合率と再現率の調和平均である．分類手法の性能は主に正確性と網羅性の質的な観点から適合率 (precision; 精度ともいう) と再現率 (recall) を用いる．適合率と再現率を処理性能の量的な観点から測定することにより判定する．適合率は分類結果として得られた集合の中にどれだけ分類に適合した文書を含んでいるかという正確性の指標であり，再現率は分類対象としている文書の中で分類結果として適合している文書（正しいで分類する文書）のうちでどれだけの文書を分類できているかという網羅性の指標である．言い換えると，適合率は，[有害文書] と分類された文書のうち，どれだけ正しく分類されているかという正確性を表し，再現率は，テストデータに含まれる有害文書をどれだけ [有害文書] と分類されているかという網羅性を表す．例えば，有害文書の適合率が低く，再現率が高い場合は，テストデータの有害文書を正しく分類していても，同時に多くの無害文書までも [有害文書] と分類していることになる．以上の理由から，本研究では有害文書の F 値を採用する．

5.2 実験設定

評価実験では，各分類手法の分類精度を示すため，実際のコミュニティサイトから収集した文書データを学習し，同様収集した文書データをテストデータとして分類する．また，既存手法であるベイジアンフィルタリングの Naive Bayes 方式，Paul Graham 方式，および小規模データベースを用いた 2 単語共起による文書分類手法との比較実験を行う事で，提案手法が既存手法に比べて優位性を持っているのか確認する．

学習データ

既存研究として，ベイジアンフィルタ（1 単語）[3][5]，及び 2 単語共起フィルタ [1] を再現するため，2 ちゃんねる掲示板から収集した無害文書を 10,000 件，有害文書を 10,000 件，合計で 20,000 件の文書を用いて，学習データの生成を行う．本研究で提案する大規模データベースを構築するために，さらに追加データとして，同様に 2 ちゃんねる掲示板から収集した無害文書 90,000 件，有害文書を 90,000 件，新しいデータセットとして 100,000 件の文書から学習データを生成した．正例と負例の文書の収集方法は 3.5 章に示した．

テストデータ

テストデータは，学習データと同様に 2 ちゃんねる掲示板から無害文書 8,000 件及び有害文書 8,000 件を収集する．また，学習データとは異なる文書を用いる．ただし，その内，10 単語以上，300 単語以下の文書のみを選択する．正例と負例の文書の収集方法は 3.5 章に示した．

5.3 実験結果

2 万件学習データを用いた実験結果は**表 7** に示す．提案手法で 20 万件学習データを用いて構築した大規模データベースを用いた実験結果は**表 8** に示す．提案手法により実装した分類手法が既存手法より高い F 値を得られることが分かった．

表 7 2 万件学習データの実験結果

手法	F 値
Naive Bayes 方式ベイジアンフィルタ (既存手法)	0.82
Paul Graham 方式ベイジアンフィルタ (既存手法)	0.81
Paul Graham 方式 2 単語共起フィルタ (既存手法)	0.81
Naive Bayes 方式ベイジアンフィルタ (提案手法)	0.83

表 8 20 万件学習データ実験結果

手法	F 値
Naive Bayes 方式ベイジアンフィルタ (提案手法)	0.93
Naive Bayes 方式 2 単語共起フィルタ (提案手法)	0.90
Naive Bayes 方式 3 単語共起フィルタ (提案手法)	0.89

6. まとめと今度の課題

本研究では，既存研究としてベイジアンフィルタと 2 単語共起フィルタの共起数を増やす，3 単語共起までのフィルタリングを実装した．特徴量 (共起) が爆発に増加したが，提案手法によりデータベースの構築が空間的，時間的に大きく改善され，3 単語共起フィルタの実験的検証を実現した．3 単語共起データベースの構築時間は既存手法と比べると，10 分の 1 に短縮した．

フィルタリングの実装手法に着目し，分類の精度に悪影響をする計算手法を差し換え，提案する BigDecimal を用いる手法では，既存手法より大規模共起を用いる際に生じる誤差で分類できない問題を解決し，大規模データベースを用いても誤差を抑える．さらに学習データを 10 倍に増加し，より大規模なデータベースを作成することで，各分類器の性能を 10%向上を実現した．

本研究で提案した手法は，ベイジアンフィルタリングと比べ，巨大な共起辞書を用いており，当然計算コストが掛かるため， データの分散が必須である．性能とコストの両面のバランスについては今後の研究の課題としたい．今回の実験では計算機 5 台にフィルタリング作業を分散することで一定の処理速度を確保することができたが，学習量を増やした場合には計算機の必要台数が急激に増加する．したがって，学習データの中から必要なデータのみを取り出す，単語のクラスタリング等で学習データをまとめる，共起の距離を制限し，データの増加を抑えるなどの工夫が必要であると考えられる．

参考文献

[1] Satoshi Ando, Yutaro Fujii, Takayuki Ito : Filtering Harmful Sentences based on Multiple Word Co-occurrence. 2010 IEEE/ACIS 9th International Conference on Computer and Information Science.
[2] Machine Learning, http://en.wikipedia.org/wiki/Machine_learning
[3] Naive Bayes classifier, http://en.wikipedia.org/wiki/Naive_Bayes_classifier
[4] BigDecimal class，Java 2 platform standard edition 5.0, API specification S Microsystem - 2004
[5] Paul Graham : Better Bayesian Filtering. Proceedings of the 2003 Spam Conference, 2003
[6] MySQL，http://www.mysql.com/
[7] InnoDB，http://www.innodb.com/
[8] MyISAM，http://en.wikipedia.org/wiki/MyISAM
[9] MeCab，http://mecab.sourceforge.net/
[10] Manning CD，Schutze H. Foundations of statistical natural lanperspectives. New York: Oxford Univ. Press, 1999. guage processing. Cambridge， MA: MIT Press, 1999.
[11] Harry Zhang ”The Optimality of Naive Bayes”. FLAIRS2004 conference.
[12] Caruana, R. and Niculescu-Mizil,”An empirical comparison of supervised learning algorithms”. Proceedings of the 23rd international conference on Machine learning, 2006.
[13] George H. John and Pat Langley (1995). Estimating Continuous Distributions in Bayesian Classifiers. Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence. pp. 338-345. Morgan Kaufmann, San Mateo
[14] 安藤哲志, 藤井雄太郎, 伊藤孝行,”有害文書判別のための多単語間共起情報辞書の構築とその応用”, 情報処理学会創立 50 周年記念全国大会第 72 回全国大会,2010
[15] 藤井雄太郎, 吉村卓也, 伊藤孝行, ベイズ分類器のスコアを素性に用いた SVM による有害文書分類手法, 情報処理学会第 74 回全国大会, 2012