学習によるXML文書のコンテンツベースフィルタリング

米井　由美† 　岩井原瑞穂†† 　吉川　正俊††

† 京都大学工学部情報学科
〒606-8501 京都市左京区吉田本町
†† 京都大学情報学研究科社会情報学専攻
〒606-8501 京都市左京区吉田本町
E-mail: †yyonei@db.soc.i.kyoto-u.ac.jp, ††{iwaihara,yoshikawa}@i.kyoto-u.ac.jp

あらまし　XMLは情報交換のための言語として多くの分野で利用されている．それに伴い，機微情報などの非公開情報のフィルタリングが重要となっている．一般にテキストやキーワードによるフィルタリングは精度の問題から人手による確認を必要としコストがかかる．XMLは部分文書や要素単位の構造による粒度の細かいフィルタリングが可能であるため，XMLの文書構造およびキーワード出現の両者を用いたコンテンツベースのフィルタリングを行う．本論文では，フィルタリングしたい内容をキーワード集合で表現し，それらを含む文書集合から，文書ベクトルおよび部分木の学習を用いてフィルタリング対象の部分文書を絞ることにより，コンテンツベースフィルタリングを行う．

キーワード　XML，フィルタリング，部分木，学習

1. はじめに

現在，データ交換に優れた性質から，XMLが情報交換のための言語として利用されることが多くなっている．患者情報や顧客情報の他，ニュースやブログ，オフィス文書, 電子商取引など様々な分野でXMLが使われている．個人のプライバシー情報や企業秘密がXML文書に格納されていることも多く，機微情報などの非公開情報のフィルタリングが重要性を増し，様々な研究が行われている．

XMLデータベースにおけるフィルタリングでは，パス条件を用いることにより，部分文書や要素単位の粒度の細かい構造によるフィルタリングが可能となっている．たとえば，顧客情報に関するXML文書において，顧客の名前のフィルタリングを行いたい場合，”//customer/name”のように，フィルタリング対象のノードをXPath式で指定することでフィルタリングが可能である．

一方，文書に出現する特定のキーワードや内容を秘匿したいという要求がある．たとえば，企業文書における新製品の情報などで，社外に対して未公開である内容や，顧客情報のnameタグ以外に出現する個人名などが挙げられる (nameタグに出現する個人名は，上記のようにXPath式を指定することでフィルタリング可能である)．しかし，このような場合フィルタリング対象は文書のタグとは無関係に出現し得るため，従来の構造のみのフィルタリングでは，テキストノード内に書かれてある内容と文書構造の間に相関性が低く，テキストの内容に対するフィルタリングに用いることができない．キーワードやテキストに対するフィルタリングは，精度に問題があるため人手による確認を必要としているのが現状である．そのためコストがかかり，膨大なデータをなるべく少ないコストでフィルタリングを行う手法に対する要求が高まっている．

そこで，本研究ではXML文書の部分文書に対するコンテンツベースフィルタリングを目的とする．

コンテンツベースフィルタリングの研究では，有害サイト[8]やスパムメール[5]を対象に，文書単位のフィルタリングが行われており，コンテンツ内に出現する単語の分布を調査し，自動的に有害か否かを判定している．単語の分布は，情報検索で用いられる tf(term frequency) と idf(inverse document frequency) を組み合わせたtf-idfをベクトルの各要素とする文書ベクトルで表現される．しかし，本研究では，文書単位ではなく比較的大きなXML文書における秘匿対象を含む部分文書をフィルタリングすることを目的としており，文書内でフィルタリング対象のみを隠す必要があるため，文書ベクトルのみの判定では不十分である．

XMLは構造化された文書であるため，パスなどの文書構造とフィルタリング対象の情報の共起を求めることにより，フィルタリングの精度を高めたり，フィルタリングする範囲を細かくする細粒度のコンテンツベースフィルタリングが可能になると考えられる．そこで，XMLの文書構造およびキーワード出現の両者を用いた細粒度のコンテンツベースフィルタリングについて検討する．本論文では，フィルタリングしたい内容を，フィルタリングキーワードと呼ぶキーワード集合で表現する．たとえば,「XXXの出身地がYYYである」という内容を隠したい場合，フィルタリングキーワードを”XXX”，”YYY”と設定する．地名”YYY”は，単独で出現する場合，機微情報とならないが，人名”XXX”と近接して共起することで，個人の出身地というプライバシー情報を示す可能性がある．そして，フィルタリングキーワードを含む文書集合に，フィルタリング対象が全て含まれると仮定する．しかしこの文書集合には対象外の

部分文書も多く含まれることも考えられる．そこで，Support Vector Machine(SVM) を用いて，文書ベクトルの学習および，フィルタリングキーワードが出現する部分文書のみを抽出し，それに対して XML の木構造の学習を行い，フィルタリング対象の部分文書を絞ることにより，コンテンツベースフィルタリングを行う．

フィルタリングで問題になることは，誤って秘匿すべき文書が公開されたり，秘匿すべきでない文書が隠されてしまうことである．そのため，精度の高いフィルタリングを行う必要がある．安全性の観点から，特に秘匿すべき文書が公開されることが問題であり，これを防ぐため，SVM による 2 値分類の際の閾値を変更することでフィルタリングの範囲を広げることを行う．さらに精度を高めるため，一般の情報検索で検索精度をユーザと対話的に改善する手法である適合フィードバックを用いる．このように，精度の高いコンテンツベースフィルタリングを実現する．

本論文では，2 章で構造によるフィルタリングと，キーワードによるコンテンツベースフィルタリングにおける関連研究を紹介する．3 章では提案手法の学習アルゴリズムとシステムの構成について，4 章では Wikipedia の XML 文書をベンチマークに実験を行った評価結果と考察について述べ，5 章でまとめる．

2. 関 連 研 究

2.1 構造によるフィルタリング

構造による XML 文書のアクセス制御ルールは以下の組からなる [1] [3].

$$(subject, object, action, decision, options)$$

subject とは，アクセスする主体のことであり，アクセス権限を与える対象である．*subject* の指定方法はいくつかあり，(1) ユーザ名，(2) ユーザグループ名，(3) ロール名，(4) サービス名などを用いることが考えられる．*object* は，アクセス制御対象を定める部分であり，URI(Uniform Resource Identifier), XPath，または両者の組み合わせで指定できる．URI によりインターネットにおける文書の格納場所が指定される．XPath を用いると，XPath を満たす XML 文書の部分木がアクセス制御対象となり，細粒度のアクセス制御が可能になる．*action* は，アクセス方法の種類であり，通常は read または write を指定する．*decision* は，grant または deny のいずれかであり，grant のときはアクセスを許可するルール，deny のときはアクセスを拒否するルールであることを示す．*options* は，ルールの適用範囲や適用方法についてのオプションであり，以下のものからなる．

- cascade: *object* で指定されたノードについて，その子孫と自身全体にルールを適用させるとき，cascade を指定する．cascade がないときは，伝播が行われず *object* で指定された対象のみにルールが適用される．
- schema: これはルールがある DTD に対応付けられており，その DTD に従う全ての文書インスタンスにそのルールが適用されるというスキーマレベルのルールであることを意味する．schema が指定されなければ，ルールがある文書インスタンスに対応付けられており，ルールはそれに対してのみ適用されるという，インスタンスレベルのルールであることを意味する．

```
<customer>
  <customer_id>123XXX</customer_id>
  <name>Alice</name>
  <address>
    <postal>
      <zip>8A3 7B2</zip>
      <street>000 City St. </street>
      <city>Tucson</city>
      <state>Arizona</state>
    </postal>
  </address>
  <phone>.123-5678</phone>
</customer>
```

図 1　XML 文書の例

図 1 のような顧客情報に関する XML 文書があるとする．

このとき，たとえば以下のようなアクセス制御ルールが作れる．

$$(customer\_desk, /customer/name, read, deny,)$$

これは，*customer_desk* が顧客の名前を読むことができないことを示す．

また，文献 [12] の Hippocratic Databases では,「データベースで貯えられる個人情報は，情報提供者から承諾がある以外の目的のために，データベースの外に伝えられるべきではない」という方針に基づいて，患者情報のデータに対して，各患者のプライバシーポリシーに従って，名前，住所，病名などを，アクセスする主体やその目的に応じて，cell 単位の粒度で開示する情報を決定している．個人情報が格納されているデータベースには，SQL でアクセスされ，各自のプライバシーポリシールールに従って，アクセスする主体，目的により，クエリーが修正され，それに応じた患者情報が返される．

2.2 キーワードによるコンテンツベースフィルタリング

コンテンツベースフィルタリングは，様々な手法で研究がなされている．

文献 [11] では，ユーザからの質問にユーザが答える知識検索サイトにおける，不適切な投稿に対するフィルタリングにおいて，人手でフィルタリングされた投稿から，フィルタリングする際に暗黙的に用いられる分類知識を表出化し，フィルタリングの自動化と分類知識の共有を試みている．ここでは，Yahoo!JAPAN の Yahoo!知恵袋を題材に，Yahoo!JAPAN がガイドラインに禁止行為として示しているもののうち,「Yahoo!JAPAN が予定していない目的で本サービスを利用している」投稿について，これらの投稿が単語間の共起頻度が低いという性質を持つことから，文章をグラフ化し，正しい文章と共起頻度を比較することによって，その重なり具合が低い場合を禁止行為とし，フィルタリングを行っている

文献 [7] では，テキストデータに含まれる人名，地名，組織

名などの固有名詞のマスキングに対して，全ての語がマスキングされている状態から，人手によって安全な語を選別してマスキングの解除を行うことで，安全性の高い機密文書のマスキングを行う手法を提案している．

3. 提 案 手 法

本研究のコンテンツベースフィルタリングの提案手法について述べる．

3.1 内容の表現方法

本論文では，フィルタリング対象の内容を複数の**フィルタリングキーワード**と呼ぶキーワード集合で表現する．たとえば，「XXX の出身地が YYY である」という内容を隠したい場合，フィルタリングキーワードを”XXX”，”YYY”と設定する．地名”YYY”は，単独で出現する場合秘匿すべき内容とならないが，人名”XXX”と近接して共起することで，”XXX”の出身地というプライバシー情報を示す可能性がある．そのため，XML 文書集合のうちキーワード集合が全て出現する文書集合を取り出すことを行う．そして，フィルタリングキーワードを含む部分文書の集合に，フィルタリング対象が全て含まれると仮定する．しかし，この文書集合には対象外の部分文書も多く含まれることが考えられる．そのため，実際にフィルタリングすべき部分文書を絞る必要がある．

3.2 学習の対象

コンテンツベースフィルタリングの学習では一般に，文書ベクトルが用いられる．しかし，本研究では XML 文書中の部分文書に対する細粒度のコンテンツベースフィルタリングを目的としているため，文書ベクトルだけでは不十分である．ここでは，フィルタリングしたい内容を複数のフィルタリングキーワードで表現している．そこで，文書中で共起するフィルタリングキーワード同士の距離が近いほど，その内容により適合すると考えられる．XML は木構造を持つため，文書内の出現位置が離れていても，親子関係にあることで関連性が強くなるとも考えられる．そこで，XML の構造を考慮した距離を学習に用いることが，細粒度のコンテンツベースフィルタリングに効果的であると考えた．用いる構造情報はフィルタリングキーワード集合が出現するテキストノードの葉までのパスを抽出した部分木である．

本論文では，文書単位ではなく，フィルタリングキーワードを包含した部分木 (**キーワード包含部分木**と呼ぶ) を単位として学習を行う．ただし文書ベクトルは，同文書に出現するキーワード包含部分木であれば，同じである．図 2 はキーワード包含部分木の生成を説明したものである．図 2 上部が XML 文書木の全体であり，フィルタリングキーワード k1，k2 が図のような位置にそれぞれ 3 個，2 個出現しているとする．このとき，フィルタリングキーワードが出現するテキストノードを葉とする部分木が，図 2 下部のように 6 個生成される．

ユーザは，キーワード包含部分木が持つフィルタリングキーワードの組み合わせに対して，文脈からその組み合わせがフィルタリングしたい内容を表現していると判断できるとき，フィルタリングすべきとする．ここで述べるユーザとは，秘匿対象

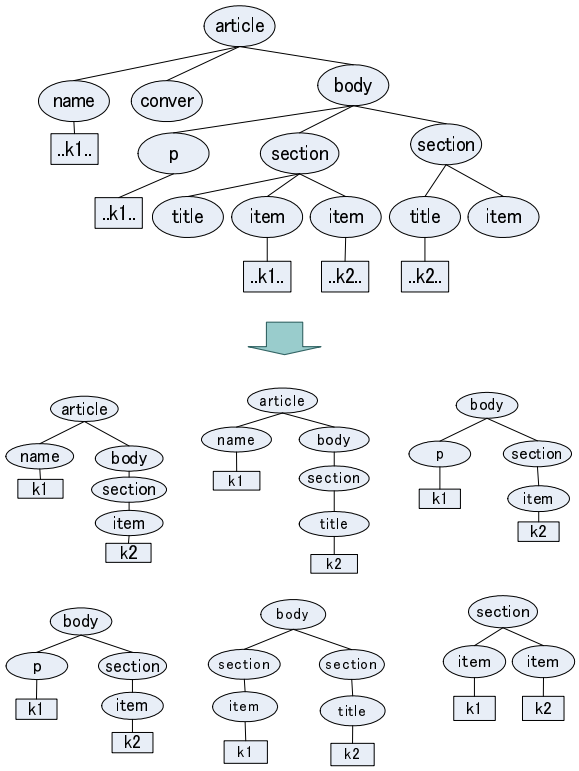

図 2　キーワード包含部分木の生成

を設定する管理者を想定しており，学習における教師の役割を行う．キーワード包含部分木を学習に用いた理由は，以下の 3 点からである．

（1） **フィルタリングキーワードの組み合わせに対する学習が可能である**

一つの文書中にはそれぞれのフィルタリングキーワードが複数回出現することがあり，それらの組み合わせによって，秘匿すべき部分文書と秘匿すべきでない部分文書がある．そのため，文書単位ではなく，フィルタリングキーワードの組み合わせに対してフィルタリングすべきかどうか判定する必要があるが，キーワード包含部分木はフィルタリングキーワード組み合わせの情報を持つため，それを可能にする．

（2） **キーワード包含部分木のサイズがフィルタリングの判定に影響する**

一般に，共起するフィルタリングキーワード間の距離が近いほど，キーワード包含部分木のサイズ (枝数) は小さくなる．本手法では，フィルタリングしたい内容を複数のフィルタリングキーワードで表現しており，文書中で共起するフィルタリングキーワード同士の距離が近いほど，その内容により適合すると考えられる．そのため，キーワード包含部分木のサイズがより小さいものほどフィルタリングすべき対象になり，サイズが大きいものほどフィルタリングすべきでない対象になる傾向がある．このように，キーワード包含部分木のサイズがフィルタリングの判定に関係する．

（3） **パス情報の学習の効果**

フィルタリングキーワードが特定のノードに出現するとフィルタリングの対象となる可能性がある．たとえば，「XXX の出身地

が YYY である」というフィルタリング内容に対して，フィルタリングキーワード”XXX”，”YYY”を設定したとする．このとき，地名”YYY”は一つの文書内に複数出現するが，address タグ内に出現する場合のみフィルタリング対象となることがある．キーワード包含部分木は，タグ名の情報を持つため，そのような学習が可能である．

以上のことから，キーワード包含部分木を学習することがコンテンツベースフィルタリングに効果があると考えられる．

図 2 のアルゴリズムでキーワード包含部分木を生成すると，フィルタリングキーワード数を多く設定したり，一つの文書におけるそれぞれのフィルタリングキーワードの出現数が多くなると，キーワード包含部分木が非常に多くなり，ユーザの学習におけるコストがかかる．このような問題を解決するために，キーワード包含部分木集合が大きくなる場合，その数を減らすことを行う．キーワード包含部分木のサイズが大きいものは，キーワード間の距離が大きく，フィルタリング対象にならない場合が多い．そこで，キーワード包含部分木が一定のサイズ (枝数) より大きく，それに完全に含まれるキーワード包含部分木が全て不正解である場合，それらを含むサイズの大きいキーワード包含部分木も不正解とし，学習の対象としない．

3.3 学 習 機 械

本研究では SVM を学習機械に用いる [2]．SVM は与えられた訓練点の中で，サポートベクトルと呼ばれるクラス境界近傍に位置する訓練点と識別面との距離であるマージンを最大化するように分離超平面を構築し，クラス分類を行う．線形分離不可能な場合は，カーネルトリックにより入力空間を線形分離可能な高次元特徴空間に写像することで，分類を行う．カーネル法は，データにアクセスする際，単体ではなく 2 つのデータの内積の形でアクセスする．この内積を与える関数はカーネル関数と呼ばれ，SVM は妥当なカーネル関数を選択することにより，高次元のデータに対しても分類を行うことができる．

カーネルのうち木構造を扱うものに，木カーネルがある [10]．二つの木，T_1, T_2 が，それぞれ V_1, V_2 の頂点集合，E_1, E_2 の枝集合を持つとき，すなわち $T_1 = (V_1, E_1), T_2 = (V_2, E_2)$ であるとき，木カーネルは次のように定義される．

$$K(T_1, T_2) = \sum_{v_1 \in V_1} \sum_{v_2 \in V_2} \sum_{s_1 \in S_{v1}(T_1)} \sum_{s_2 \in S_{v2}(T_2)} K^S(s_1, s_2) \quad (1)$$

$$K^S(s_1, s_2) = I(s_1 = s_2) \quad (2)$$

ここで，$S_v(T)$ は $v \in V$ を根に持つ部分木の集合を表し，K^S は二つの部分木の間に定義されるカーネル関数であるとする．式 (2) の $I()$ は，括弧内が成立する場合に 1，そうでない場合に 0 となる関数である．また，$s_1 = s_2$ は，2 つの部分木が完全一致するとき真となる．このように部分構造を用いて再帰的に木カーネルが定義される．

木構造の学習は，自然言語処理の分野における構文解析木 [13]，バイオインフォマティクスの分野における RNA の木構造データ，HTML や XML などの Web データで行われている．

このように SVM は，木カーネルを用いることでキーワード包含部分木に対しての学習を行うことができる．また，文書ベクトルは，一般に次元数が非常に大きいが，SVM は大きい次元数に対応できる．さらに，一般にユーザの評価できるデータ数は少ないが，SVM は少ない訓練例からの学習に適しているという性質がある．以上のことから学習機械に SVM を利用した．

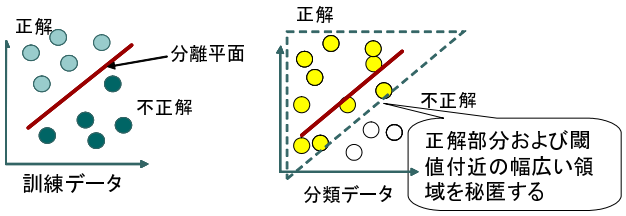

図 3　閾値の考察

3.4 訓練データの選択方法

パターン認識問題では，学習済みのパターンから何らかの法則性・規則性を見つけ出し，未学習のパターンにそれを適用し，正しい分類を得るという汎化能力がある．しかし，特定のパターンの訓練データのみの学習を行うと，それに対して系が収束し過ぎるため，その他の入力に対する汎用性がなくなってしまう過学習が生じる．そのため訓練データは，正解，不正解に対して多様なパターンを含む必要がある．

一般に，共起するフィルタリングキーワード間の距離が近いほど，キーワード包含部分木のサイズ (枝数) は小さくなる．本手法では，フィルタリングしたい内容を複数のフィルタリングキーワードで表現しており，文書中で共起するフィルタリングキーワード同士の距離が近いほど，その内容により適合すると考えられる．そのため，キーワード包含部分木のサイズがより小さいものほどフィルタリングすべき対象になり，サイズが大きいものほどフィルタリングすべきでない対象になる傾向がある．

そこで，訓練データ集合と分類データ集合を比較したとき，それらが同程度のサイズのキーワード包含部分木のデータを持つように，全ての文書からランダムに訓練データを選択する．これによって，訓練データ集合，分類データ集合ともに，様々なサイズのキーワード包含部分木のデータを持ち，正解，不正解に対して多様なパターンを含むことができる．

3.5 閾値の考察

フィルタリングでは，誤って秘匿すべき文書が公開されてしまうことおよび秘匿すべきでない文書が隠されてしまう可能性を低くする必要がある．そのため，精度の高いコンテンツベースフィルタリングを行う必要がある．安全性の観点から特に誤った公開が問題であり，それを防ぐことを行う．

学習結果に基づいて，分類データを SVM で正解 (フィルタリングする) と不正解 (フィルタリングしない) の 2 値に分類する際に，不正解と分類されたが分離平面付近に存在するものは，安全のため秘匿する方が望ましい (図 3)．そこで，2 値分類の際の閾値の変更を行い，フィルタリングの範囲を広げ，そのようなデータを秘匿することを試みる．

SVM は学習でユーザにより，正解 1，不正解− 1 のスコア

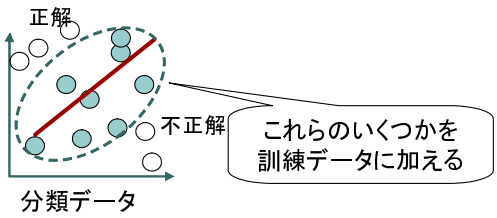

図 4　適合フィードバック

が与えられる．これをもとに，分類データに対してスコアが計算され，一般に 0 を閾値として，0 以上が正解，0 未満が不正解に分類される．この閾値を 0 より小さい値に変更し，フィルタリングを行う範囲を広げることで，秘匿すべき文書が誤って公開されることを防ぐ．

3.6　適合フィードバック

一般の情報検索において，検索精度をユーザと対話的に改善する手法として，適合フィードバック (relevance feedback) がある．この手法は，提示された検索結果に対し，ユーザが適合，非適合の判定を行い，その判定結果をシステムにフィードバックして，さらに適合性の高い文書を検索する．本研究に対して，適合フィードバックを応用することで，学習結果を改善することを行う．

分離平面から離れた距離にある分類データは，訓練データとの類似性が高く学習により正しく分類されている可能性が高い．一方，そうでない分類データは，訓練データとの類似性が低く学習により正しく分類されていないことがある．そこで，これらの分類データを訓練データに加えることで学習済みの結果をシステムにフィードバックさせ，学習を行うことにより，最初の学習に含まれなかったパターンのデータに対しても学習を行うことができ，さらに適合性の高い結果になると考えられる (図 4)．

そこで，分類データの中から，SVM によって与えられたスコアの絶対値が設定値より小さいもののいくつかを訓練データに加え学習を行い，学習済みの結果をシステムにフィードバックさせる．

3.7　システム構成

本研究におけるフィルタリングシステムは, 主に以下の 5 段階から構成される (図 5)．

step1　**フィルタリングキーワード集合で検索を行う**

ユーザがフィルタリングしたい内容をキーワード集合で表現し，システムが対象となる XML 文書集合からそれらのフィルタリングキーワードを全て含む文書の検索を行う．

step2　**検索結果の文書集合から，キーワード包含部分木集合を生成する**

システムが文書集合から図 2 のアルゴリズムでキーワード包含部分木を抽出し，キーワード包含部分木集合を生成する．

step3　**キーワード包含部分木集合に対して，ユーザを教師とする学習を行う**

ユーザはそれぞれのキーワード包含部分木に対して，それが出現する文書の文脈からフィルタリングすべきかどうかの判定を

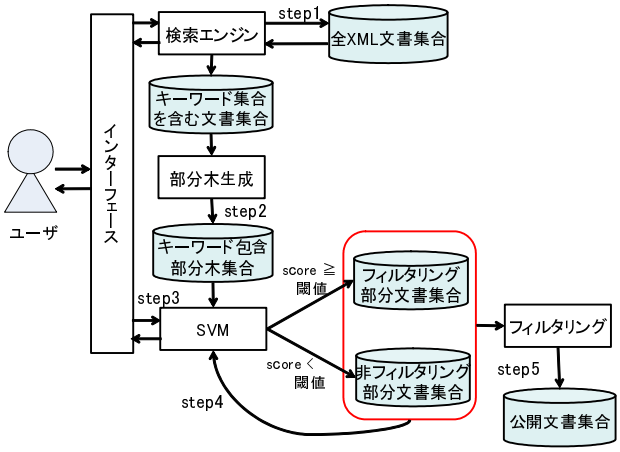

図 5　システム構成図

行う．すべきとき正解とし，すべきでないとき不正解とする．訓練データ集合と分類データ集合を比較したとき，それらがサイズが同程度のキーワード包含部分木のデータを持つように，システムが全ての文書からランダムに訓練データを k 件選択し，ユーザがそれらに正解不正解を与えることで学習を行い，残りを学習結果に従って分類する．

step4　**適合フィードバックを行う**

システムは分類データで SVM により計算されたスコアの絶対値が設定値より小さいデータのいくつかを訓練データに加えて学習を行い，学習済みの結果をシステムにフィードバックさせる．スコアが閾値以上のデータをフィルタリング対象とする．

step5　**フィルタリングを行う**

学習結果を反映して，システムはフィルタリング対象の部分文書集合にフィルタリングを行う．

4.　実　　験

4.1　実 験 環 境

4.1.1　ベンチマーク

実験のベンチマークとして Wiki で作られたフリー百科事典である Wikipedia を用いる(注1)．Wikipedia は XML ではないが，構造化された文書であり，2002 年から INEX プロジェクトにおいて Wikipedia の XML 化が行われており [4]，Wikipedia の DTD も作成されている．入手のしやすさからこれを選んだ．2006 年 3 月 3 日現在の日本語版の文書には 78,158 件の文書が含まれている．

Wikipedia は公開されている文書であるため，著作権やプライバシー侵害等で削除された書き込みを除いて，機微情報は存在しないと考えられる．そこで，個人のプロフィールが記述されている文書に書かれている個人情報などを，秘匿すべき情報と仮定して例題を作成した．また，Wikipedia は文書内に多くのリンクを持つ．リンク部分のテキストは，木の深さが 1 つ深くなり，サイズが大きくなるが，意味的距離はリンク部分以外のテキストと同等である．そこで，本研究ではリンクによる構造記述を無視してキーワード包含部分木を作成する．

(注1)：http://ja.wikipedia.org

4.1.2 検索エンジン

3.7 の step1 で XML 文書からキーワード集合を含む文書集合を取り出す検索エンジンには，XML データベースに対するキーワード検索システムである Kikori-KS [14] を用いた．一般に索引語の重みは，tf(term frequency) と idf(inverse document frequency) を組み合わせて利用する tf-idf を用いることが有用であるが，Kikori-KS では XML 文書の構造を利用したより有効な指標である ipf(inverse path frequency) を用いている [6]．要素 E 中に出現する索引語 t の重み $weight(t, E)$ は以下のように定義される．

$$weight(t, E) = \frac{ntf}{nel} * ipf \tag{3}$$

$$ntf = 1 + \ln(1 + \ln(tf)) \tag{4}$$

$$nel = ((1 - s) + s * \frac{el}{avgel_p}) * (1 + \ln(avgel_p)) \tag{5}$$

$$ipf = \ln \frac{N_p + 1}{ef_p} \tag{6}$$

ntf は正規化された索引語出現回数 (tf) であり，nel は要素長を反映した正規化指標である．また，ipf は経路中の索引語の特定性である．ここで，el は要素中の索引語の数，p は E の根ノードからの経路，$avgel_p$ はある経路 p に存在する要素の el の平均値，Np は p に存在する要素の数，ef_p は経路 p に存在する要素の内 t が出現するものの数である．また，s は定数のパラメータであり，ここでは 0.2 を用いる．

文書ベクトルを求めるために，Kikori-KS で計算された索引語の重み $weight(t, E)$ の値を用いた．

4.1.3 SVM

学習機械は，SVM^{light} を用いた(注2)．Vladmir N. Vapnik が考案した SVM を実装したもので [15]，最適化アルゴリズムは Thorsten Joachims の手法を採用している [9]．大きなデータセットを高速に処理可能であり，設定ファイル kernel．h において新規のカーネルを定義することができる．カーネルがいくつか用意されており，その中に木カーネルがある(注3)．これは，自然言語処理などの木構造学習に用いられる．また，木とベクトルを組み合わせた学習も可能であり，部分木と文書ベクトルを組み合わせた学習に SVM^{light} の木カーネルを用いた．

4.2 評　　価

SVM により分類されたデータに対して精度と再現率の評価を行う．キーワード集合を含む文書集合のうち，実際にフィルタリングされた文書集合を A，フィルタリングすべき部分文書集合を B とする．このとき，精度 (precision) と再現率 (recall) は以下のようになる．

$$\text{精度} = \frac{|A \cap B|}{|A|}$$

(注2)：http://svmlight.joachims.org/

(注3)：http://ai-nlp.info.uniroma2.it/moschitti/TK1.2-software/Tree-Kernel.htm

例題 1 「中村俊輔選手の出身地が横浜市である」: 中村俊輔，横浜
例題 2 「安倍晋三首相の出身地が山口県である」: 安倍晋三，山口
例題 3 「高原直泰選手の出身地が静岡県である」: 高原直泰，静岡
例題 4 「シャープが携帯電話を開発している」: シャープ，携帯
例題 5 「東芝が DVD を開発している」: 東芝，DVD

図 6　例　　題

表 1　実験データ

	例題 1	例題 2	例題 3	例題 4	例題 5
検索文書数	7	9	8	11	16
包含部分木数	42	42	66	100	152
正解数	8	9	11	47	79
不正解数	34	33	55	53	73
訓練データ数	20	20	30	50	70
分類データ数	22	22	36	50	82

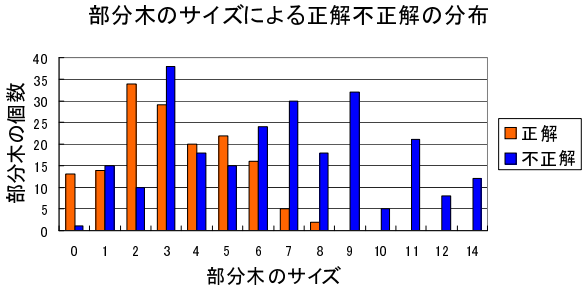

図 7　キーワード包含部分木のサイズと正解不正解の関連

$$\text{再現率} = \frac{|A \cap B|}{|B|}$$

精度は実際にフィルタリングされた部分文書がフィルタリングされるべき部分文書である割合を表し，再現率はフィルタリングされるべき部分文書が実際にフィルタリングされた割合を表している．本研究のフィルタリングにおいては，フィルタリングすべき部分文書がフィルタリングされないことが問題となるため，精度よりも再現率を重視する．

4.3 実験結果と考察

実験は図 6 にある 5 つの例題に対して行った．それぞれの例題は政治家やスポーツ選手などのプライバシー情報や企業の技術情報を，秘匿すべき情報と仮定して作成した．表 1 は実験データに関する情報を示している．

それぞれの例題は，訓練データ集合と分類データ集合を比較したとき，それらが同程度のサイズのキーワード包含部分木のデータを持つように，全ての文書からランダムに，表 1 の訓練データ数のデータを選択し，それに対して学習を行い，その結果をもとに残りのデータを分類した．

図 7 は全ての例題の正解データと不正解データにおける，キーワード包含部分木のサイズによる分布を示している．グラフより，正解となるデータは木のサイズが小さくなる傾向があり，サイズ 9 より大きいデータが存在しないことが分かる．一方，不正解データは，木のサイズが幅広い範囲に存在している．

表 2　異なる訓練データに関する実験結果

		訓練データ 1	訓練データ 2	訓練データ 3	平均
例題 1	精度	1.0	1.0	1.0	1.0
	再現率	1.0	1.0	0.75	0.92
例題 2	精度	0.67	1.0	1.0	0.89
	再現率	1.0	0.40	0.40	0.60
例題 3	精度	0.80	0.75	0.86	0.80
	再現率	0.80	0.50	1.0	0.77
例題 4	精度	0.52	0.44	0.40	0.45
	再現率	0.54	0.60	0.50	0.55
例題 5	精度	0.87	0.79	0.79	0.82
	再現率	0.95	0.95	0.93	0.94

そのため，部分木のサイズだけで正解と不正解を分離できず，文書ベクトルや部分木のパス情報の学習も重要となる．

学習に用いる訓練データの選択が学習効果に及ぼす影響を確認するため，それぞれの例題に対して，異なる訓練データを用いて実験を行った．また，3.5 で述べたように，フィルタリングでは秘匿すべき文書が誤って公開されることが問題であるため，秘匿すべき文書の範囲が広がるように，閾値の変更に関する実験を行った．さらに，3.6 で述べた方法により適合フィードバックを行い，最後に閾値の変更と適合フィードバックの両者を組み合わせた実験を行った．

4.3.1　異なる訓練データに関する実験

表 2 はそれぞれの例題に対して，3 つの異なる訓練データで実験を行った結果である．ここで，閾値は 0.0 であり，適合フィードバックは行っていない．

実験の結果，全ての例題に対する平均精度，平均再現率はそれぞれ 79% ，75% であった．しかし，例題によっては評価が 5 割に満たないものもあり，良い結果ではなかった．例題 2 は木のサイズが小さくても不正解，大きくても正解になるデータが存在し，それらのデータに正しく学習が行われなかったため，訓練データにより結果の良し悪しが大きく異なった．例題 4 は，精度，再現率ともに 5 割程度という悪い結果を得た．これは，不正解データの木のサイズが幅広く存在しており，訓練データに含まれる不正解データの個数が多く，SVM により分類データに対して計算されたスコアが負の値に偏ったものとなったからである．例題 1, 3 は文書によって正解と不正解の部分木のサイズがきれいに分かれており，文書ベクトルおよび部分木の学習が効果的であった．また，例題 5 は文書によってフィルタリングキーワードが特定のノードに出現するときに正解となることが多く，文書ベクトルおよびパスの学習が効果的であった．そのため，これらは安定して良い結果を得た．

4.3.2　閾値の変更に関する実験

秘匿すべき文書が誤って公開されてしまうことを防ぐため，閾値を負の値に変更し，フィルタリングの範囲を広げることを行った．閾値を 0.0, -0.1, -0.2, -0.3, -0.4 と変更し比較実験を行った (表 3)．実験結果は異なる訓練データの平均の値であり，適合フィードバックは行っていない．

閾値を変更することにより，閾値-0.4 のとき，平均再現率が 84% と高くなったが，平均精度は 65% と低い値となった．また，例題によっては閾値を変更しても変化の無いものや，再現率は変化しないが精度が低下するものがあった．

表 3　閾値の変更に関する実験結果

	閾値	0.0	-0.1	-0.2	-0.3	-0.4
例題 1	精度	1.0	0.83	0.83	0.83	0.50
	再現率	0.92	0.92	0.92	0.92	0.92
例題 2	精度	0.89	0.89	0.89	0.89	0.78
	再現率	0.60	0.60	0.60	0.60	1.0
例題 3	精度	0.80	0.80	0.80	0.80	0.80
	再現率	0.77	0.77	0.77	0.77	0.77
例題 4	精度	0.45	0.45	0.45	0.45	0.45
	再現率	0.55	0.55	0.58	0.58	0.58
例題 5	精度	0.82	0.82	0.75	0.73	0.71
	再現率	0.94	0.94	0.95	0.96	0.96
平均	精度	0.79	0.76	0.75	0.74	0.65
	再現率	0.75	0.75	0.76	0.76	0.84

表 4　適合フィードバックに関する実験結果

適合フィードバック	平均精度	平均再現率
なし	0.79	0.75
あり	0.89	0.89

表 5　閾値の変更と適合フィードバックの組み合わせに関する実験結果

適合フィードバック	閾値	0.0	-0.1	-0.2	-0.3	-0.4
なし	精度	0.79	0.76	0.75	0.74	0.65
	再現率	0.75	0.75	0.76	0.76	0.84
あり	精度	0.89	0.89	0.89	0.79	0.86
	再現率	0.89	0.90	0.94	0.95	0.95

4.3.3　適合フィードバックに関する実験

適合フィードバックに関する実験を行った．分類データのうち，SVM により与えられたスコアの絶対値が 0.8 以下のものの中から 10 個程度を訓練データに加え学習を行い，学習済みの結果をシステムにフィードバックすることで，精度，再現率ともに高めることを行う．表 4 は適合フィードバックを行った場合と行わなかった場合の比較実験である．ここで，閾値は 0.0 であり，実験結果は全ての例題の平均の値である．

実験結果から，適合フィードバックにより，平均精度，平均再現率が 89% , 89% となり，ともに向上しており，また，全ての例題において平均の評価が上がっていた．特に，例題 4 は適合フィードバックを行わないとき，平均精度，平均再現率が 45%, 55% と低い値であったが，適合フィードバックを行うことで平均精度，平均再現率が，70% , 91% となり，良い結果を得た．これは最初の学習で訓練データに含まれなかった正解データが，適合フィードバックにより訓練データに加えられ，正解データに対する汎化能力が高められたからである．以上のことから，適合フィードバック手法が有効であったことが分かる．

4.3.4　閾値の変更と適合フィードバックの組み合わせに関する実験

表 5 は閾値の変更と適合フィードバックの組み合わせに関する実験結果である．閾値の変更では，閾値を 0.0, -0.1, -0.2, -0.3, -0.4 と変更しており，適合フィードバックでは，分類デー

タのうち，SVM により与えられたスコアの絶対値が 0.8 以下のものの中から 10 個程度を訓練データに加え，学習済みの結果をシステムにフィードバックすることを行っている．実験結果は全ての例題の平均の値である．

閾値の変更と適合フィードバックを組み合わせることにより，閾値-0.4 のとき，平均精度が 86% ，平均再現率が 95% であった．精度は閾値の値を小さくすることで悪くなっていたが，適合フィードバックを行うことで精度の減少が抑えられた．これは，最初の学習では訓練データとの類似性が低いが，適合フィードバックにより訓練データに加えられたデータと類似する残りのデータが正しく学習され，全体のスコアの絶対値が大きくなり，閾値の変更により不正解とすべきデータがフィルタリングされてしまうことを減少することができたためである．

5. おわりに

本研究では，XML の部分文書に対するコンテンツベースフィルタリングを行った．細粒度のフィルタリングを行うために，文書ベクトルおよび部分木の学習を用いてフィルタリング対象の部分文書を絞る手法を提案した．また，フィルタリングでは誤って秘匿すべき文書が公開されることおよび秘匿すべきでない文書が隠される可能性を低くする必要があり，安全性の観点から特に誤った公開が問題である．そこで，SVM の 2 値分類の際に閾値を変更し，フィルタリングの範囲を広げることを行い，さらに精度を高めるため，分類データにおいて SVM で計算されたスコアの絶対値が設定値より小さいデータのいくつかを，訓練データに加えて学習を行い，学習済みの結果をシステムにフィードバックすることで，精度の良いフィルタリングを試みた．

本手法の有効性を確かめるため，Wikipedia の XML 文書をベンチマークに，5 つの例題を作成し実験を行った．SVM において過学習を防ぐため，訓練データ集合と分類データ集合を比較したとき，それらが同程度のサイズのキーワード包含部分木のデータを持つように，全ての文書からランダムに訓練データを選択することを行った．異なる訓練データに関する実験を行ったところ，訓練データにより大きく影響される例題もあったが，安定して良い結果を得た例題もあった．閾値の変更に関する実験では，例題によっては再現率は変化せず，精度が低くなるものがあったが，平均再現率は向上した．また，適合フィードバックを行うことで，精度，再現率ともにより良い結果を得た．さらに，閾値の変更と適合フィードバック手法を組み合わせることにより，閾値-0.4 のとき，平均再現率は 95% まで向上し，平均精度は 86% となりともに高くなり，特に再現率が良い結果を得た．

以上の実験結果から本手法が XML の部分文書に対するコンテンツベースフィルタリングに有効であることが確認できた．

今後の課題として，SVM だけでなく複数の学習機械についての検討が挙げられる．また，ベンチマークを Wikipedia としたが，その他の XML 文書に対しても同様にフィルタリングが可能か実験を行いたい．さらに，今回，木のサイズに枝数を用いているが，根と葉の関係と兄弟の関係を区別し，キーワードの分布状態により，精密に木のサイズを定義する必要がある．

謝辞 Kikori-KS の日本語化で協力いただいた京都大学情報学研究科の清水敏之氏に心より感謝致します．

文　献

[1] E. Bertino, S. Castano, E. Ferrari, M. Mesiti, ”Specifying and Enforcing Access Control Policies for XML”,Document Sources, WWW Journal, Vol.3, Vo.3, 2000.

[2] Nello Cristianini, John Shawe-Taylor 著, 大北剛 訳, ”サポートベクターマシーン入門”, 共立出版

[3] E. Damiani, S. De Capitani di Vimercati, S. Paraboschi,P. Samarati, ”A Fine-Grained Access Control System for XML Documents,” ACM TISSEC, Vol. 5, Vo. 2, 2002.

[4] Ludovic Denoyer, Patrick Gallinari,”The Wikipedia XML corpus”,ACM SIGIR Forum Vol.40 No.1 June 2006.

[5] H. Drucker, eith C. Wu and V. Vapnik, ”Support Vector Machines for Spam Categorization”, IEEE Trans. On Neural Networks, Vol.10, No.5, pp. 1048-1054. 1999.

[6] Liu, F., Yu, C. T., Meng, W. and Chowdhury, A., ”Effective keyword search in relational databases”, Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data, Chicago, USA, pp. 563-574. 2006.

[7] 伊川洋平，宅間大介，金山博，”安全語のアンマスキングによる機密情報のマスキングシステム”，電子情報通信学会, 2006.

[8] 井ノ上直己, 帆足啓一郎, 橋本和夫, ”文書自動分類手法を用いた有害情報フィルタリングソフトの開発”, 電子情報通信学会論文誌 D-ⅡVol. J84-D-Ⅱ No. 6 pp. 1158-1166 June 2001.

[9] Thorsten Joachims, ”Learning to Classify Text Using Support Vector Machines”, Dissertation, Kluwer, 2002.

[10] 鹿島久嗣，坂本比呂志，小柳光生，”木構造データに対するカーネル関数の設計と解析”，人工知能学会論文誌 Vol.21, No.1,a, 2006.

[11] 小林大祐，松村真宏，石塚満，”知識サイトにおける有害情報のフィルタリング知識の表出化”, The 20th Annual Conference of the Japanese Society for Artificial Intellifence, 2006

[12] Kristen LeFevre, Rakesh Agrawal, Vuk Ercegovac, Raghu Ramakrishnan, Yirong Xu, David DeWitt, ”Limiting Disclosure in Hippocratic Databases”, Proceeding of the 30th VLDB Conference, Toronto, Canada, 2004.

[13] Alessandro Moschitti, ”Making Tree Kernels proctical for Natural Language Learning”,EACL 2006.

[14] Toshiyuki Shimizu, Norimasa Terada, and Masatoshi Yoshikawa, ”Kikori-KS: An Effective and Efficient Keyword Search System for Digital Libraries in XML”, 9th International Conference on Asian Digital Libraries (ICADL 2006), Lecture Notes in Computer Science (LNCS), Springer-Verlag, Vol. 4312, pp. 390-399, Kyoto, Japan, November 27-30, 2006.

[15] Vladimir N,Vapnik, ”The Nature of Satisitical Learning Theory”, Springer, 1995.