制約付クラスタリングを用いた
適合性フィードバックに関する研究

（Relevance feedback using constrained clustering）

辻下 卓見

東京大学情報理工学系研究科電子情報学専攻
担当教官　安達 淳

2008 年 2 月提出

目次

図 目 次

表 目 次

第1章　はじめに

1.1　研究の背景

World Wide Web(WWW) 上に流通するデータは、近年のブログや WIKI などの使いやすいツールの普及や、様々な WEB サービスの隆盛によって、爆発的に増大しつつある (図 1.1)。そのような膨大な情報の中から目的を持って情報を得ようとする時、多くのユーザは Google や Yahoo などといったキーワード検索型の検索サービスを利用する。

しかしキーワード検索ではユーザが入力する質問単語数は平均二、三語であるため、望むものが検索結果の上位に上らない場合も多い。その場合、ユーザは何らかの手段で検索結果の絞り込みを行う必要がある。現在使われている Web 検索サービスでは絞り込みはユーザによるキーワードの追加で行われるが、適切な結果を得られるキーワードを入力できるかどうかはユーザの検索能力によるところが多い。インターネット世帯浸透率が 80 %をこす現在 [1]、すべてのユーザに検索能力を求めるのは適当ではなく、システム側が検索を容易にすることが必要とされている。

こういった問題を解決するための手法は多数提案されているが、それらの手法は大きく二つに分類することができる。

一つはユーザの過去の行動を記録してユーザの嗜好を抽出し、検索結果を向上させる手法である。この手法はパーソナライズド検索と呼ばれ、様々な手法が提案され実装されている [14][28]。もう一つはユーザとの対話を通じて、ユーザの嗜好を読み取り、検索結果を改善するやり方である。対話を通じた検索によって良い結果が得られることは、Koenemann と Belkin の研究によって証明されている [17]。ユーザとのやり取りの方法としては、検索結果をクラスタリングして提示する Clusty[2] や、適切な拡張質問を示して絞り込みを助ける手法 [4] など様々な方法が提案されている。そのような手法の一つとして、検索で得られた結果から適合する文章と適合しない文章をユーザから入力してもらうことで検索結果を改善する適合性フィードバックがある。適合性フィードバックは、検索結果が適合す

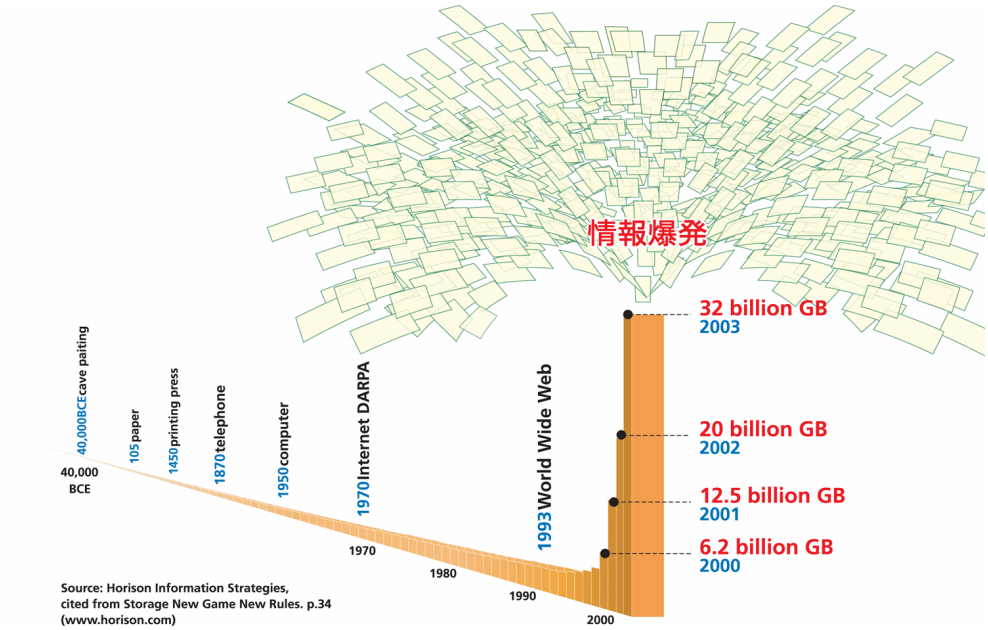

図 1.1: WEB 上の情報の爆発的な増大

るかしないかを人手や自動で判定した結果を利用して検索結果を改善する手法である.

適合性フィードバックは従来型の情報検索 [15][18] や、Web を対象とした情報検索 [26][30] でも良い結果が得られている。また、適合性の判定は検索結果に付随する文書要約を見ればできる場合が多いため、Web 検索エンジンにしかし一方で少数の例をのぞけば、現在のところ適合性フィードバックを使った一般的な検索サービスは普及していない。それは、良い結果が得られるのは多数の適合性判定情報を得たられているときであり、通常の検索においてはユーザから十分な量の適合性判定情報を得られることはまれであるからである [9]。すなわち、ユーザの労力と性能の適切なトレードオフがとれていないため、実用的なサービスとして普及していないといえる。また、適合性フィードバックのインターフェースが複雑になるからという指摘もある [3]。

1.2 研究の目的

適合性フィードバックの既存の手法では、検索結果を改善するのにユーザの入力を多数必要とする。そこで本研究では、適合性フィードバックの手法を改善し、より少数のユーザ入力で、良いフィードバック結果を得る手法を提案することを目的とする。

具体的には、近年研究の進んでいる教師付クラスタリングの一つである、制約付クラスタリングを用いる。制約付クラスタリングは、通常クラスタリングでは事前知識を利用することなく処理を行うのに対して、ユーザが持っている背景知識をインタラクションを通じて得ることで、それを制約としてもちいてクラスタリング精度を上げる技術である。

既存手法に用いられている適合性フィードバックで用いる文書の適合性判定情報は制約付クラスタリングの制約に変換することができる。そこで提案手法では、以下のようなプロセスで適合性フィードバックを行う。

まず、ユーザからの検索クエリを基に検索結果を返す。この検索は、既存手法で確立されている手法を用いる。この検索結果に対して、ユーザが適合文書と不適合文書のどちらか、あるいは両方をいくつか判定する。それによって与えられた適合文書と不適合文書から制約付クラスタリングで用いる must-link、cannot-link という二種類の制約条件を用意する。次に制約付きクラスタリングを用いて、用意した制約条件を満たしたクラスタ集合を作る。クラスタリングの結果として得られたクラスタの中から適合クラスタと不適合クラスタは擬似的な適合文書と不適合文書の集合として考えることが出来る。すなわちユーザによる適合性判定情報が増えたと考えることが出来る。そしてその擬似的な適合性判定情報を用いて、検索質問の修正を行い、検索結果を改善する。この過程はユーザが満足する結果を得るまで繰り返される。

提案手法では一般的な検索サービスである Google と Yahoo などのインターフェースを基に自然に拡張されたインターフェースを想定している。

1.3 論文の構成

第一章以下、本論文は以下のような内容で構成される。

- 第 2 章
  適合性フィードバックに関する関連研究
- 第 3 章
  本研究の提案手法の概要と、提案手法を校正する細かい要素手法について
- 第 4 章
  実世界データを使った実験による提案手法の有効性の評価
- 第 5 章
  実験結果の評価と考察、今後の課題について
- 第 6 章
  結論

第2章　関連研究

本章では適合性フィードバックについての一般的な概念と、既存研究について述べる。

2.1 適合性フィードバック

本節では、適合性フィードバックについて簡単に説明する。適合性フィードバックは、ユーザとのインタラクションを通じて、検索結果を改善していく手法である。

一般的な検索エンジンでは、検索はユーザによる検索クエリ入力、検索システムからの結果の表示、というプロセスで行われる。しかし、実際には入力される検索クエリは短いもの [13][27]、あるいは曖昧ものである場合 [23][19] が多い。ユーザが自身の情報欲求を適切な形で検索クエリに変換するひつようがあるためである。

また、ユーザが変われば同じクエリでも適合する文章は異なってくる。たとえば、生物学者が ”mouse”という検索クエリを入力した場合、ネズミに関する情報を必要しているが、プログラマーが同じクエリを入力している場合は、コンピュータの入力デバイスであるマウスに関する情報が必要になる。

よって情報検索においては、一度の検索で必要とする情報が得られるとは限らない。そこで検索結果を一度提示し、ユーザがそれを見て検索システムの結果を適切なものに出来るようシステムのパラメータを調整することが考えられる。このような技術を一般に適合性フィードバック (Relevance Feedback) と呼ぶ。その際、ユーザからどのような情報を得るか、その結果、何を変更するかが重要な要素となる。

適合性フィードバックでユーザ入力としてもちいる代表的なものとしては、検索結果の適合性判定情報がある。検索結果として得られた文書ランキングから適切な数の文書をユーザに提示し、それらについてユーザに適合かどうかを判定させる。その情報をもちいることで、検索結果の適合するもの、適合しないものを得ることが出来る。

適合するものを用いた場合、正のフィードバック、適合しないものを用いた場合負のフィードバックとなる。また、両方を用いる場合もある。

適合性フィードバックでは、多くの場合ユーザからの情報を使って検索質問を修正する。修正の対象としては他に文書と検索モデルがあるが、ともに長期的に修正の影響が及ぶため、必ずしも好ましいとは限らない。ユーザごとにプロファイルを作成し、文書や検索モデルに修正を加える方法はパーソナライズドサーチと呼ばれ、研究される場合が多い。

ユーザからの適合性判定情報をつかって検索質問を修正する一般的な適合性フィードバックの概略図 2.1 である。

次節以降では具体的な適合性フィードバックの手法についていくつか説明する。

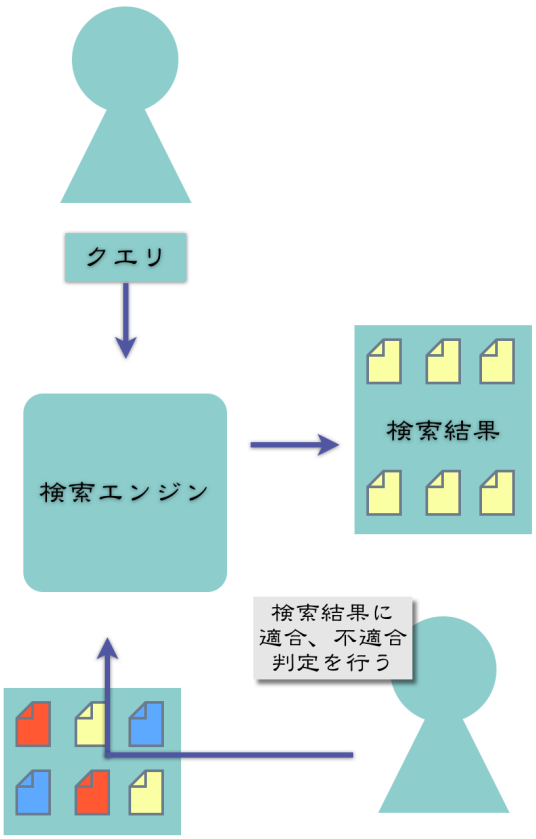

図 2.1: 適合性フィードバック

2.2 学習なしフィードバック

本節ではユーザから得た適合性判定情報をつかって検索質問を修正するときに、機械学習などの技術を用いない手法、特に様々な手法で使われることの多い、ベクトル空間モデルを使った Rocchio の手法と、疑似適合性フィードバックにおいて多用される Robertson の手法、及びその他いくつかの手法について述べる。

2.2.1 Rocchio の手法

Rocchio の手法はベクトル空間モデルで、ユーザの適合判定情報から検索質問を修正することでフィードバックを行う手法である。

情報検索の代表的なモデルの一つであるベクトル空間モデルでは、文章と検索クエリを単語の重み付きベクトルとして表現する。単語の重みには文書中での出現回数や、より一般的には TFIDF などが使われる場合が多い。検索時には、検索クエリを単語ベクトルに変換した後、文書すべての単語ベクトルごとに、クエリのベクトルとの類似度を計算し類似度のランキングを結果としてユーザに返す。類似度としてはベクトルの余弦を用いる場合が多い。

ベクトル空間モデルにおける適合性フィードバックでは、ユーザの適合性判定情報から得られた適合文書、不適合文書を用いてクエリの単語ベクトルの重みを修正する。そして新しいクエリを用いて検索をし直す。単語ベクトルの修正する手法はいくつか提案されているがいずれも次式が基本となっている。

$$Q_{n+1} = Q_n + \alpha \sum_{j=1}^{N^+} D_j^+ - \beta \sum_{j=1}^{N^-} D_j^-$$

式において、Q_n は n 回目の検索質問しおける検索クエリの単語ベクトルである N^+, N^- はそれぞれ n 回目の検索における適合文書、不適合文書の数であり、D_j^+, D_j^- はそれぞれその時の適合文書、不適合文書の単語ベクトルである。α, β はパラメータであり、α が大きい場合は適合文書を、β が大きい場合は不適合文章それぞれどの程度重要視するかと云うことを調整する。Q_{n+1} はフィードバックによって得られる新しい検索クエリの単語ベクトルである。

この式では適合文書に含まれる単語の重みを大きく、不適合文書に含まれる単語の重みを小さくするよう検索クエリの単語ベクトルを変更する。

Rocchio の手法では上式のパラメータ α と β をそれぞれ $1/N^+, 1/N^-$ と置いたものである。これは適合性フィードバックにおける適合文書と不適合文書の重要度

を文書の数によって正規化したものである。

$$Q_{n+1} = Q_n + 1/N^+ \sum_{j=1}^{N^+} D_j^+ - 1/N^- \sum_{j=1}^{N^-} D_j^-$$

Rocchio の手法は適合性フィードバックの様々な手法の中でもっとも基本となるものの一つであり、様々な研究で用いられている。

Rocchio の手法の具体例をあげる [32]。

表 2.1: Rocchio の手法の例

$Q_1 : (3, 6, 7, 2, 2, 7)$
$+d_1 : (1, 4, 3, 1, 1, 3)$
$-d_2 : (4, 1, 3, 6, 7, 1)$
$+d_3 : (2, 4, 2, 2, 4, 2)$
$+d_4 : (3, 1, 2, 3, 4, 2)$
$-d_5 : (5, 1, 1, 4, 4, 1)$
$Q_2 : (0.5, 8, 7.3, -1, -0.5, 8.3)$

たとえば Q_1 というクエリに対して得られた検索結果が表のような単語ベクトルだったとする。ここで d_1, d_3, d_4 は適合文書と d_2, d_5 が不適合文書とユーザによって判定されたとする。このとき、Rocchio の手法によって得られる新しいクエリ Q_2 は $(0.5, 8, 7.3, -1, -0.5, 8.3)$ となる。この Q_2 をもちいて再度検索し直すことで、適合性フィードバックが完了する。

一般に検索結果不適合文書の数は適合文書の数に比べて多いことと、ユーザが判定できる数は検索結果全体から見るとごく一部であるため、信頼性の高い不適合文書の情報を利用することが難しい場合がある。このような場合を考慮して、不適合文書の情報は最上位を用いる Ide dec-hi の式など、検索対象に応じた Rocchio の手法から派生する式もある。

$$Q_{n+1} = Q_n + \sum_{j=1}^{N^+} D_j^+ - D_1^-$$

2.2.2 Robertsonの手法

Robertsonの手法は、適合文書の情報を使って適合性フィードバックを行う手法である[24]。Robertsonの手法では、ユーザが判定した適合文書に含まれる単語に適合度に関するスコアを与えてランキングを作り、ランキングの上位の単語を検索クエリに追加するという手法を用いている。Robertsonの手法で各単語に与えるスコアは次の式で与えられる。

$$wpq_t = (\frac{r_t}{R} - \frac{n_t - r_t}{N - R}) \times \log \frac{\frac{r_t}{(R-r_t)}}{\frac{n_t-r_t}{(N-n_t-R+r_t)}}$$

ここで、r_tは単語tを含む適合文書数、n_tは単語tを含む文書数、n_tは単語tを含む文書数である。また、Rは既知の適合文書数、Nはコレクション内の文書数である。第二項目はRobertson/Spark Joneの重みと呼ばれ[25]、Okapi検索システムの単語の重み付けにも利用されている。

このスコア付けは通常の検索システム以外にも、クラスタリングの結果からラベルを抽出する場合など、様々な場で利用される。また、検索システムに用いる場合は、疑似適合性フィードバックで用いられることも多い。

疑似適合性フィードバック

疑似適合性フィードバックとは、ユーザからの判定情報を用いることなく適合性フィードバックを行う手法である。通常の適合性フィードバックでは、検索結果の上位に対してユーザが適合文書か不適合文書かを判定して、その判定情報を用いて当たらし結果を得るが、疑似適合性フィードバックでは検索結果のランキング上位にきているものは、適合していると考えてフィードバックを行う。Robertsonの手法を用いた疑似適合フィードバックは以下のように行われる。まずユーザの検索クエリを使って検索結果のランキングを得る。ランキングの上位N件を適合文書と考え、それらの文書に含まれる単語に対して、スコアを計算する。スコアの上位の単語を検索クエリに加えることで、検索結果の改善を行う。この手法はTRECにおける大規模検索実験によって検索精度が平均的には向上することが示されているが、検索課題毎に調べると、4分の1くらいについてはフィードバック適用前よりも検索有効性が下がってしまうというように、初期検索の性能に結果が大きく依存してしまうという問題がある。

2.2.3 その他の手法

適合性フィードバックは他にも様々な手法が提案されている。

Robertson & Sparck Jones

確率モデルを用いた検索モデルでは、文書が検索クエリに適合しているかという確率を求めることで、検索結果を得る [24]。しかしそのためには検索クエリに対する適合文書がすべて求まっている必要がある。これは現実的ではないため、適当な値でこれらの値を推定し、検索結果をユーザに判定させ、それを適合する文書のサンプルとして用いることでパラメタを推定するという適合性フィードバックが行われる。確率モデルを用いた適合性フィードバックは、ベクトル空間モデルにおけるフィードバックに比べて理論的によりしっかりしているという利点がある。

他には検索要求をクエリとして表現することが困難な画像検索では様々な手法が提案されているある。たとえば、ベイズ理論を利用してユーザの検索要求を推定しながら結果を改善していく、Bayesian feedback algorithm[5] という手法などがある。

Scatter / Gather

検索結果をクラスタリングしてから表示し、それを用いる適合性フィードバックとして Scatter / Gather が有名である [6]。Scatter / Gather は動的なクラスタリングを通じて文書を絞り込む手法である。まず検索結果文章が決められた数のクラスタに分類され、各クラスタに対してラベルが付けられる。ユーザはそれをもとに適合しているクラスタ群を選択する。クラスタ群は併合され、もう一度分類される。以上を繰り返すことで、文章集合を絞り込んでいうプロセスでフィードバックを行う (図 2.2)。

WEB-GRAPH RERANKER

これまでで述べた手法は、主に新聞などの文章コレクションを対象として提案された手法であるが、Web を対象とした適合性フィードバックの手法もまた提案されている。Web では通常の文書分類と異なり、より多くの情報を用いることが

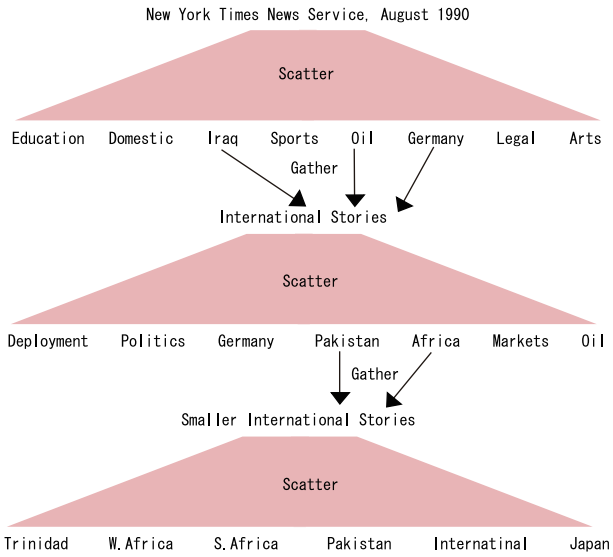

図 2.2: Scatter / Gather

できる。Sergei らは文書の内容ではなく、リンク構造を用いることで、より WWW に適した適合性フィードバックの手法 WEB-GRAPH RERANKER を提案している [29]。この手法では、PageRank や HITS と同様に適合ページは適合ページを、不適合ページは不適合ページとリンクしているという考え方を用いている。ユーザは通常の適合性フィードバックと同様に、検索結果に対して適合、不適合の判定を行う。次に WEB-GRAPH RERANKER は WEB のリンク構造に従って検索結果の URL 同士を結ぶグラフ構造を作る。そして適合文書から閾値以内の距離にある文章に適合度を加えていく。不適合文層についても同様に行う。そして最終的な適合度の大きさに応じて文書をランキングし直す。

また、前節で述べた Rocchio の手法や Robertson の手法、Bayesian feedback algorithm などの手法を WEB 文書に適用した研究もある。

2.3 学習ありフィードバック

本節では、フィードバックを行うときに、機械学習などの技術を用いてユーザの負担を減らして実用性を高めている手法について述べる。

2.3.1 SVM を用いた手法

Support Vector Machine(SVM) は精度が高いことで知られ、近年多くの研究が進んでいる機械学習技術である。本節では SVM を使って適合性フィードバックの精度をあげている手法について述べる。

Support Vector Machine

SVM は二クラスの分類問題に対応した線形分類器の一つであり、汎化能力が高いため学習データ以外に対しても良い結果が得られる。また学習や分類が高速であることから多くの問題に対して用いられている。

SVM では、学習データの正例の集合と負例の集合の間に、マージンが最大になるように超平面を決定する。マージンとは超平面に最も近い正例、負例と超平面の間の距離のことである。二次元平面上の問題を例とすると図 2.3 のようになる。分類したいデータが与えられたとき、超平面を挟んで正例側にあるか負例側に存在するかでデータは分類される。

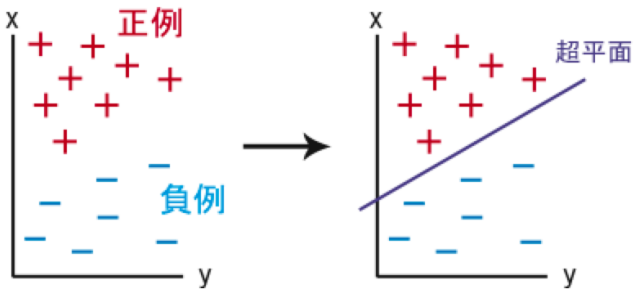

図 2.3: Support Vector Machine

SVM の学習フェーズである超平面の決定は、マージンを最大にするという問題になり、これは

$$\frac{1}{2}\|\vec{w}\|^2 \qquad subject\ to\ y_i(\vec{w}\cdot\vec{x_i}-b)\geq 1, \forall i$$

という式を最小化する最適化問題とみなすことが出来る。このとき x_i は i 番目の学習データであり、y_i は i 番目の学習データに対する正しい SVM の出力である。この最適化問題を解くことで、超平面の重みベクトル $\vec{w}$ が求まる。

実際に分類したいデータ $\vec{x}$ が与えられたとき、$\vec{x}\cdot\vec{w}$ を計算してその正負を判定することで分類する。

上記の例で用いた線形 SVM では、学習データが超平面で分離できない非線形な問題を解くことは出来ないが、非線形 SVM ではカーネルトリックと呼ばれる手法を用いて分類したいデータを高次元に写像するため、このようなデータも分離可能である。

通常の SVM は二クラス分類であるが、複数クラスの分類問題を解くために SVM を拡張した、Multiclass-SVM という手法もある。

Drucker らの手法

Drucker らの手法では、SVM を用いることで、適合性フィードバックを行っている [8]。この手法では、ユーザから得られた適合性判定情報を基にして、SVM の学習を行う。ただしこのとき、通常の SVM で用いられる w のかわりに、Q^* を用いる。このときマージン $\frac{2}{|Q^*|}$ は次の式を最大化する α を見つけることで最大化される。

$$W(\alpha)=\sum_{i=1}^{N}\alpha_i - 5\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j(D_i\dot{D}_j)y_iy_j$$

このとき $0\leq\alpha_i\geq C, y_i=\pm 1, \sum y_i\alpha_i=0$

$$Q^*=\sum_{i=1}^{r}\alpha_i y_i D_i$$

である。

つぎに、SVM によって求めた Q^* を使ってランキングをやり直す。まだ出てきていない文書 D_i 全てに対して、Q^*D_i を計算する。そしてこの値が大きかったものはより適合するものであるとしてランキングし直す。すなわち分離超平面から

適合文書側に離れていれば離れているほど適合であり、不適合文書側にいくほど不適合となりランクが下がっていく。

以上の処理を満足できる結果が得られるまで繰り返す。

Onoda らの手法

SVM は二クラス分類器であるため、学習データとして正例と負例を与えなければ SVM を構築することが出来ない。よって、適合性フィードバックを行う場合も初期検索の上位に適合文書と不適合文書の両方がなければ、実行することが出来ない。しかし、一般的に適合文書の数は不適合文書のそれに比べて非常に少ないため、ランキングの上位に適合文書がこない場合もある。

そこで Onoda らはランキング上位に適合文書がない場合でも適用可能な、不適合文書だけで適合性フィードバックを行う手法を提案している [22]。

ランキングの上位に不適合文書しかない場合、学習には不適合文書のデータしか与えられない。そういった一つのクラスしか学習データが与えられない場合に適用可能な、One-Class SVM や SVDD といった SVM を拡張した機械学習の手法がある。

One-Class SVM はカーネル関数を使うことで、一クラス問題に対応する分離超平面をつくることで、SVDD は分離超球面をつくることで一クラスの分類器を作る (図 2.4)。フィードバックは以下のプロセスで行われる。

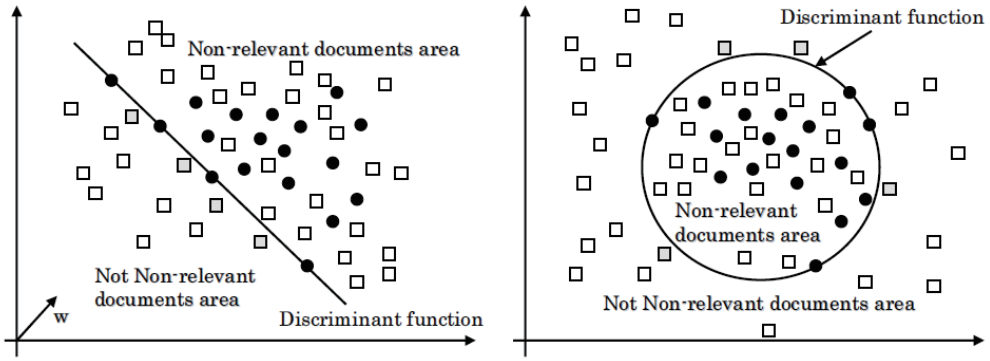

図 2.4: One-Class SVM と SVDD

1. まず初期検索の結果として得られるランキングから、トップ N 個の文書を選択する。

2. 次にトップN個の文書からから不適合文書を選択する。得られた不適合文書の情報を基に、One-class SVM か SVDD を用いて分類器を生成する。

3. 最後に、生成した分類気を遣ってトップN個の文書のうち選択されていないものを分類し、不適合文書と判定されなかったものを、再びユーザに提示する。

新聞記事を用いた実験の結果では、One-class SVM の方が一貫して良い結果が得られている。

2.3.2 Transductive 学習を用いた手法

機械学習をつかって適合性フィードバックを行う場合、学習に用いるデータと、分類したいデータはともに検索結果であるため、通常機械学習で通常重要視される汎化性能は、求められない。新しいデータを分類する必要がないため、分類器を生成する必要がない。そこで、岡部らは Transductive 学習と呼ばれる機械学習と Robertson の手法を用いた適合性フィードバック手法を提案している。

Transductive 学習

Transductive 学習は transduction と呼ばれる推論方法を基にした訓練データから分類器を生成することなく調節テストデータのラベル付けを行う機械学習である。

Transductive学習を実現するアルゴリズムは、これまでに様々なものが提案されており、様々なタスクで学習データが少ない場合での優位性が示されている。Okabe らはそのなかで高い性能を持つことが示されている Spectral Graph Transducer (SGT) アルゴリズムを用いている。SGT はテストデータ集合へのラベル割り当てを ratiocut 問題として定式化して、緩和問題を解くことによって近似解を導き出すアルゴリズムである。Transductive 学習を二次元空間上に存在するデータに対して適応した場合の概念図は図 2.5 のようになる。

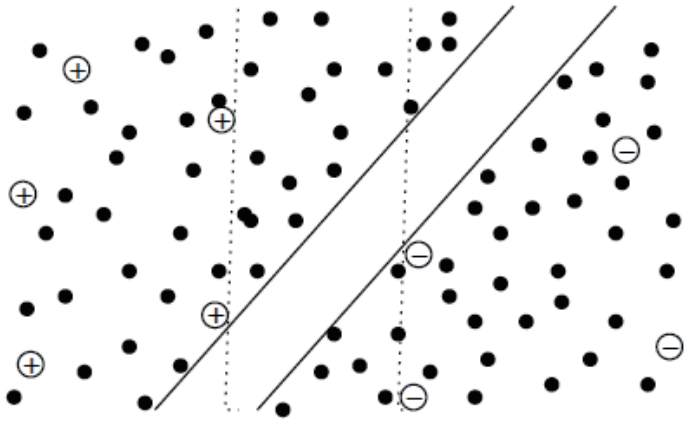

図 2.5: Transductive 分類

Onoda らの手法

Onoda らは上記の SGT を用いて以下のような流れでクエリ拡張を行っている。

1. 初期検索：検索システムへの初期クエリの入力
2. 適合性判定：訓練データ用の文書として、初期検索の結果の上位から適合文書と非適合文書を一ずつ指定する。
3. 暫定的適合文書の発見：図 2.6 に示すように SGT を用いることで他に適合している可能性のある文書を発見する。暫定的適合文書とは学習アルゴリズムが適合文書と判断した文書である。
4. クエリ拡張のための単語選択：Robertson の式と、SGT によって得られる文書が適合である可能性の値を用いて、各単語のスコアを計算し、スコアが高いもの上位 m 件を初期クエリに追加する単語として選択する。
5. 拡張クエリによる再検索：初期クエリと全ステップで得られた追加単語を拡張クエリとして再び検索を行う。

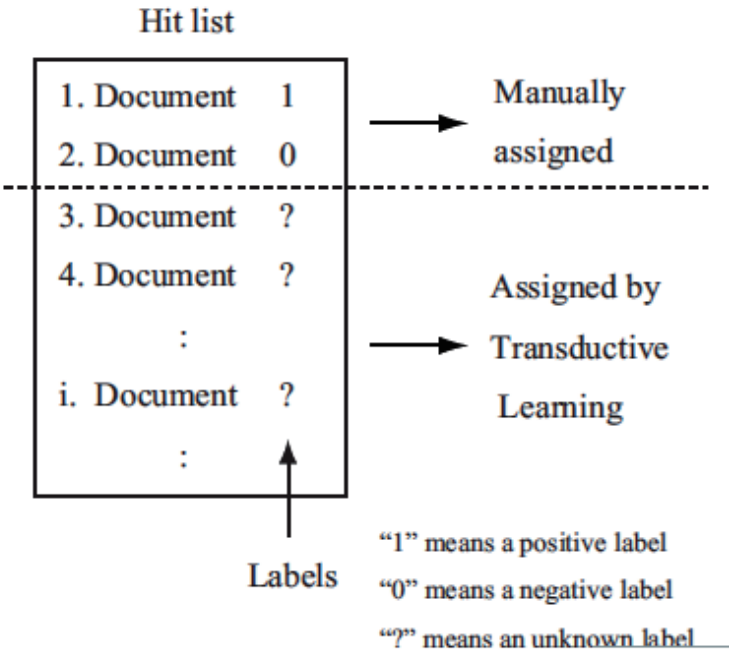

図 2.6: 適合文書の発見

2.4 手法の比較

適合性フィードバックは以上のように様々な手法があるが、実用サービスとして使うにはそれぞれ一長一短である。

Robertson や Rocchio の手法では、十分な量の適合性判定情報が得られなければ良い結果は得られないため、ユーザの負担が大きい。疑似適合性フィードバックを行う場合は、初期検索の性能に依存するという問題がある。また、ユーザのフィードバックがないため、ユーザによってことなる情報要求に応じた結果を出すことが出来ない。

Scatter / Gather は検索結果が見やすいなどの利点もあるが、クラスタリング時にユーザの意図を反映できないという問題がある。また、検索システムのインターフェースが特殊であるため、ユーザになじみづらいという問題も考えられる。

一方で Robertson や Rocchio の手法の十分な量の適合性判定情報が必要であるという問題の対策として提案されている、機会学習を用いて適合性フィードバックの精度を上げる手法にも問題が考えられる。SVM や Transductive 学習を用いる場合、確かに非常に少数の学習例からフィードバックを行うことが可能であるが、適合情報と不適合情報の両方が得られなければ学習を行うことが出来ない。

１クラスの学習器を用いる場合、不適合情報だけから適合性フィードバックを行うことは可能であるが、適合情報と不適合情報の両方が入力された場合、別の手法にスイッチしてフィードバックを行う必要がある。

よってどれも適合情報と不適合情報のどちらか、あるいは両方が適合性判定情報としてくるという三つの場合を統一的に扱えないという問題がある。

そこで本研究では、適合性判定情報が少数な場合でもそれら三つの場合を統一的に扱う手法を提案する。

表 2.2: 適合性フィードバック手法の比較

	Robertson など	Okabe の手法	Onoda の手法	提案手法
必要な適合性情報	多量	少量	そこそこ	少量
適合文書と不適合文書	適合文書	両方	不適合文書	両方又は片方

第3章　提案手法

本章では本研究の提案手法の概要と、提案手法を実現するために用いている、検索手法や制約付クラスタリングについて述べる。

3.1 提案手法の概要

本論文では、既存研究の適合性フィードバックにおける、ユーザによる多量の適合性判定情報を必要とする問題や、適合情報と不適合情報を統一的に扱うことが出来ない問題に対応した手法を提案する。

提案手法では通常の適合性フィードバックと同様に、まず質問を検索エンジンになげて検索結果を得る。その中のいくつかの文書を適合文書か不適合文書かに振り分ける。そしてその適合性判定情報をもとに、検索結果を改善する。その際、ユーザとのインタラクションを行うインタフェースとして、図 3.1 のようなものを想定している。このインターフェース Google や Yahoo などの WWW を利用するユーザに親和性が高いと考えられる

図 3.1: 提案手法のインターフェース

提案手法をより具体的に述べると、ユーザによる初期検索の結果に対する、適合性判定情報を制約という形に変換し、制約付クラスタリングを行うことで、 ユーザによって適合性判定を受けていない文書に対して擬似的に適合性判定を行う。擬似的な適合性判定を行うことで、適合文書と不適合文書の数が増えるため、より良いフィードバック結果を得ることが出来る。

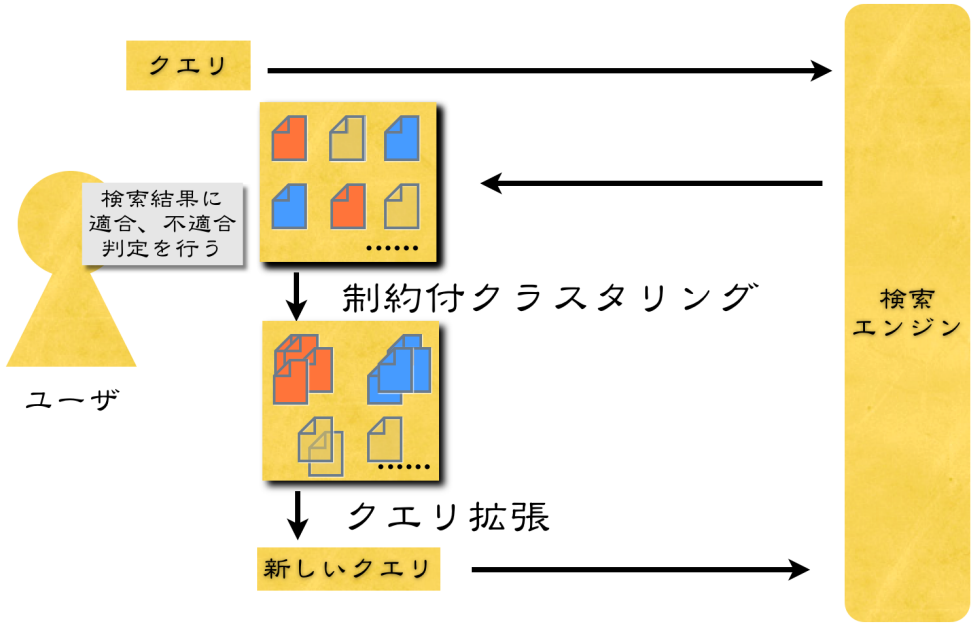

図 3.2: 提案手法

提案手法は以下のようなプロセスで実行される。

- **Step 1**:ユーザが入力した質問を既存の検索エンジンに投げた結果上位 N 件を表示する。
- **Step 2**:検索結果 N 件のうち適合文章と不適合文書を人手でいくつか判定する。判定した結果から must-link と cannnot-link という二種類の制約条件を生成する。
- **Step 3**:得られた制約条件を付けて、検索結果の上位 N 件を制約付きクラスタリングという手法で分類する。その結果として、適合文書クラスタ、不適合文書クラスタ集合、その他の集合が得られる。
- **Step 4**:適合文書クラスタ、不適合文書クラスタ集合を用いて、通常の検索質問の修正手法にかけることで、新たな質問が得られる。その質問を用いて検索し直すことでよりよい検索結果が得られる。

提案手法の概略は図 3.2 になる。

提案手法では、検索機能の実現方法、制約付クラスタリングの手法に幾つかの手法について採用し、実験で比較、検討した。

次節以降では提案手法の詳細について述べる。

3.2 検索機能の実現

適合性フィードバックを用いたシステムを構築するためには、通常の文書検索システムを実現しなければならない。本研究では、検索モデルとして文書検索に一般的に用いられるベクトル空間モデルを採用した。本節では、ベクトル空間モデルと単語の重み付け、単語ベクトルの類似尺度について述べる。

ベクトル空間モデル

文書検索を行う場合、何らかの形で、文書の言語表現から意味を抽出する必要がある。しかし現在の技術では言語表現の意味を完全に抽出することは難しいため、文書の内容をよく表していると考えられる要素を抽出し、その集合で文書の内容を表現するということが一般的に行われる。このような文書の内容を表す要素には索引語と呼ばれる単語が使われる。索引語は、単語から検索に用いるのに適さないストップワードと呼ばれる語などを除いたものである。

文書から抽出された索引語に対して、その索引語の文書における重要度を表す尺度を与えることを単語の重み付けと呼ぶ。重み付けを行うことで、同じ索引語を含む文書でも、その索引語の各文書中の重要度を考慮して、検索質問に対する文書の適合度を計算することで、検索結果となるランキングを得ることが出来る。

$$
\begin{array}{c} \\ t_1 \\ t_2 \\ t_3 \\ t_4 \end{array}
\begin{array}{c}
\begin{array}{cccc} d_1 & d_2 & d_3 & d_4 \end{array} \\
\begin{pmatrix} 0 & 1 & 2 & 0 \\ 3 & 4 & 2 & 0 \\ 0 & 0 & 0 & 3 \\ 5 & 0 & 2 & 1 \end{pmatrix}
\end{array}
$$

図 3.3: 単語-文書行列の例

次節で後述する手法で各文書中の索引語に対して重み付けを行うと、図 3.3 のような行列が得られる。この行列において、各行は索引語を、各列は各文書に対応する。また、行列要素 $a_{i,j}$ は索引語 t_i の d_j における重みを表す。

ここで行列の各列を列ベクトルと見なすと、列ベクトルは文書を表す単語ベクトルであると見なすことが出来る。また、同様に検索クエリも単語の重み付けベ

クトルとして表現することが出来る。

検索対象の文書と検索クエリをともに単語で表現することで、各文書の検索質問への適合度をベクトル間の類似度に帰着することが出来る。このようにベクトル間の類似度によって検索質問に対する文書の適合度を計算する検索モデルをベクトル空間モデルと呼ぶ。

ベクトル空間モデルでは検索の前にあらかじめすべての文書を単語の重みベクトルに変換する。また、検索時には検索クエリも単語の重みベクトルに変換し、クエリのベクトルとすべての文書のベクトルとの類似度を計算し、類似度の高い順にユーザに文書を提示する。

3.2.1 単語の重み付け

前節でも述べたとおり、ベクトル空間モデルで検索を行う場合、各文書ごとに単語の重要度を示す重みを計算しなければならない。単語の重みを考える際には、一般的に網羅性と特定性という二つの性質を考慮しなければならないとされる。

網羅性は検索クエリに適合する文書をもれなく抽出するという性質であり、特定性とは正しい文書だけを抽出するという性質である。一般に特定性を高くするには、その文書には現れるが、他の文書には現れない単語を重視すればいい。しかし文章に特有の単語を重要視しすぎると、検索質問でその単語が使われる可能性も低くなる。一方で一般によく使われる語を重要視すると、多くの文書が検索されることになる。しかし、検索された文書がユーザの必要とするものであるとは限らなくなる。このように特定性と網羅性は背反な関係にあり、両者のバランスをとる必要がある。すなわち、網羅性と特定性を兼ね備えた単語の重要度が上がるように重みを付ける必要がある。

TF-IDF

TF-IDF は網羅性を考慮した重みである TF(term frequency) と、特定性を考慮した IDF(Inverse Document Frequency) を組み合わせ、両者の性質を同時に考慮した重み付け手法である。TF-IDF はもっとも一般的な重み付け手法の一つで、多くのベクトル空間モデルの検索システムで用いられている。

TF とはある文書 d に出現する単語 t の頻度であり、$tf(t,d)$ であらわす。TF に基づく重み付けは「何度も繰り返し言及される概念は重要な概念である」という仮定がある。しかし、一般に頻度の高すぎる単語は、文章を特徴づける上ではあまり役に立たない。また、文章が長くなると平均的に語の出現頻度も多くなる傾向にある。そのため頻度をそのまま重みとして採用してしまうと、同じ単語でも長い文書に現れる場合の方が重みが大きくなってしまう。そのため、本研究では、次のように単語の出現頻度を文章中のすべての単語の出現数で割った相対頻度を用いている。

$$w_t^d = \frac{tf(t,d)}{\sum_{s \in d} tf(s,d)}$$

TF では単語の網羅性を高めることには貢献するが、特定性には役に立つとは限らない。TF は各文書の中での頻度は考慮していても、文書集合内での他の文書の

頻度などは考慮していないためである。単語がどの文書にどの程度特徴的に現れるかどうかは、他の文書での単語の頻度分布も考慮する必要がある。たとえば、どの文書にも高頻度に出ている単語よりは、ある文書に特徴的に高頻度に出ている単語の方が大きな重みを与えられるべきである。

IDF はこのような特定性を表すために、ある単語が全文書中のどれくらいの文書に出現するかを表すの尺度であり次の式で定義される。

$$idf(t) = \log \frac{N}{df(t)} + 1$$

ここで N は全文書数、$df(t)$ は単語 t が出現する文書数（文章頻度）である。式からわかるように、IDF はある索引語が少数の文章に現れる場合に大きな値をとる。

TF-IDF は TF と IDF の両者を掛け合わせることで、特定性と網羅性を同時に考慮している重み付けであり、次式で表される。

$$w_t^d = tf(t,d)\dot{i}df(t)$$

bm25 TF-IDF

BM25 は確率的な考え方をもちいた検索モデルであり、高い性能が得られることがわかっている [20]。

本研究では、Robertson-Sparck Jones の重みを TF-IDF で用いられる IDF の log で置き換えたものを用いる [11]。この置き換えによって、性能が改善されるという研究結果が知られている。本研究で用いる BM25 の、文書 d における単語 t の重み w(t) は以下の式で与えられる。

$$w(t) = log(\frac{N}{df}) \cdot \frac{(k+1) \cdot tf}{k\{(1-\alpha) + \alpha \frac{dl}{avdl}\} + tf}$$

ここで、N は文章集合中の全文書数。df は単語 t の文章頻度、dl は文書 d の文書長。$avdl$ は平均文書長。k, α はそれぞれコントロールパラメータである。

ここでは、既存研究に従って、パラメータをそれぞれ $alpha = 0.5$、$k = 20$ に固定している。

3.2.2 類似尺度

ベクトル空間モデルを用いた場合、実際の情報検索は、前項などの手法を用いて単語に重み付けを行った、検索クエリの単語ベクトルと文書の単語ベクトル間の類似度によって検索結果を求める。全文書と検索クエリとの間の類似度を計算し、類似度の大きさによってランキングを作成する。

また、類似尺度は後述するクラスタリングを用いる際、クラスタリング対象となる文書同士の類似度を求める際にも用いられる。

ここでは、一般的に用いられることの多い、コサイン類似度と、距離関数であるユークリッド距離、Kullback-Leibler 距離を用いた Jensen-Shannon エントロピーについて述べる。

コサイン（余弦）類似度

コサイン（余弦）、文章検索においてもっともよく用いられる類似尺度である。コサインは２つのベクトルのなす角度を表しているため、値が１である時にもっとも類似していることになり、値が０のときは２つの文章は全く異なっている。コサインは次式で定義される。

$$\begin{aligned} cos(d_x, d_y) &= \frac{d_x \dot{d}_y}{\|d_x\| \, \|d_y\|} \\ &= \frac{\sum_{i=1}^{n} d_{xi} d_{yi}}{\sqrt{\sum_{i=1}^{n} d_{xj}^2} \sqrt{\sum_{i=1}^{n} d_{yj}^2}} \end{aligned}$$

ここで、d_x, d_y はそれぞれ類似度を求めたい単語ベクトル、n はベクトルの長さである。クエリと文書の類似度を求めたい場合、d_x, d_y のどちらかがクエリに対応する単語ベクトルになる。たとえば以下のような三つの三次元ベクトルを考える。

$$\begin{aligned} v_1 &= (1, 2, 3), \\ v_2 &= (100, 200, 300), \\ v_3 &= (2, 2, 2) \end{aligned}$$

このときコサイン類似度では、v_1 と v_2 の類似度は 1、v_2 と v_3 の類似度は０ 0.926 になる。

コサインは単語ベクトルの長さが１に正規化されている場合、コサイン尺度と

内積 $d_x\dot{d}_j = \sum_{i=1}^{n} d_{xi} \cdot d_{yi}$ は一致する。クラスタリングは計算量が高くなる傾向にあるため、あらかじめ文書を正規化して計算量を減らす場合もある。

検索では類似度として他に Dice 係数、Jaccard 係数などが用いられる。

Dice 係数

$$\frac{2d_x \cdot d_y}{\sum_i d_{xi}^2 + \sum_i d_{yi}^2}$$

Jaccard 係数

$$\frac{d_x \cdot d_y}{\sum_i d_{xi}^2 + \sum_i d_{yi}^2 - d_x \cdot d_y}$$

ユークリッド距離

ユークリッド距離は通常の空間で定義される距離と同じであり、以下の式で計算される。

$$d(d_x, d_y) = \sqrt{\sum_i (d_{xi} d_{yi})^2}$$

ただし、情報検索の分野では、単語ベクトルの次元は数万に行くことも珍しくないため、ユークリッド距離を用いることはほとんど存在しない。

本研究では、後述するクラスタリングの手法の一つにおいて文章間の類似度に距離を用いなければならないため、簡単な距離関数の一つとして用いた。

Jensen-Shannon エントロピー

単語の重み付きベクトルが単語の確率分布であるとすると、２つの文書間の類似性の判定は、２つの分布の類似度の問題と考えることが出来る。情報処理の分野では分布間の類似度を求める基準として Kullback-Leribler 情報量がある。２つの分布 $P = \{p_1, p_2, ..., p_M\}$ と $Q = \{q_1, q_2, ..., q_M\}$ がある時、これらの分布間の Kullback-Leribler 情報量は次式で表される。

$$\begin{aligned} KL(P\|Q) &= E_p \log \frac{P}{Q} \\ &= \sum_{i \in M} \log \frac{p_i}{q_i} \end{aligned}$$

しかしKullback-Leribler情報量には、q_iの値が0のときに無限大になるため、単語ベクトルのようにスパースなデータを扱う場合にはスムージングを行わなければならないという性質がある。

また、式から見てわかるようにPから見たQとのKullback-Leribler情報量とQから見たPとのそれでは値が変わってしまうという非対称性がある。そのため、いわゆる距離の公理を満たさないという問題がある。

そこでこういったKullback-Leribler情報量の問題を解消したものとしてJensen-Shannonエントロピーがある[12]。Jensen-Shannonエントロピーは Kullback-Leribler情報量を用いているが、確率０への耐性と、対称性を持つように定義されている。

$$JS(P,Q) = \frac{1}{2}\left[KL(P \| \frac{P+Q}{2}) + KL(Q \| \frac{P+Q}{2})\right]$$

3.3 クラスタリング

本節では、クラスタリングについての概要と、具体的なクラスタリング手法について述べたあと、クラスタリングに制約を付けるよう修正した制約付クラスタリングの各手法について述べる。

クラスタリングは様々なデータマイニングで用いられる技術であり、データを類似性に基づいてグループ化させる手法の一つである。クラスタリングでは入力として与えられたデータ集合に対して、それぞれ共通の特徴を持つような部分集合に分割する。

データ集合を部分集合に分類するという問題では、あらかじめ分類される先となるラベルを用意し、それぞれのラベルについて学習データを用意し機械学習の技術を用いて分類器を構築し、それを利用することで分類することが多い。

一方クラスタリングはそのような一般的に用いられる教師付き機械学習とよばれる手法とは異なり外的な基準なしに自動的に分類する手法である。すなわち分類先のラベルや学習データをあらかじめ用意する必要がない、非教師付機械学習である。

あらかじめラベルを用意する必要がないため、機械学習のように学習データを用意するコストがいらないが、一方で高い性能を出すためにはデータセットに対してパラメータのチューニングする必要性もある。

クラスタリングによって得られるクラスタについて文献 [10] では、クラスタを「内的結合と外的分離が達成されるような部分集合」と定義を試みてはいるが，形式的な定義な困難であるとも同時に述べている．すなわちクラスタリングは望ましいクラスタの基準を形式的に与えることが困難であるという問題がある。そこで背景知識を妥当なクラスタの基準として使う様々なやり方が提案されてきている。

クラスタリングの手法は、似たもの同士を併合していくつかの グループにまとめて行く階層的なクラスタリング (hierarchical clustering method) と、似たものが結果的に同じグループに入るように集合を分割する非階層的クラスタ リング (non-hierarchcal clustering method) とに大きくわけて考えることが出来る。

3.3.1 階層的クラスタリング

階層的クラスタリング (凝縮型) は、1 個の対象だけを含む N 個のクラスタがある初期状態から、クラスタ間の類似度に基づき、最も類似度の大きな二つのクラスタを順に併合して一つのクラスタにしていくという操作を繰り返すことで、クラスタリングを行う手法である。クラスタリングの結果は、図 3.4 のような木構造になる。この木構造をデンドログラムと呼ぶ。

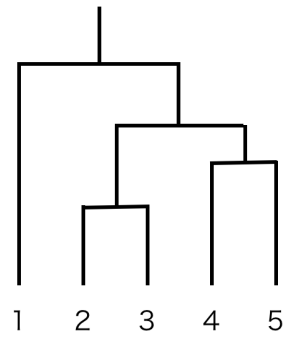

図 3.4: デンドログラム

クラスタリングのアルゴリズムは以下の三つの手続きからなっている。

1. 初期化
   (a) 各要素だけからなるクラスタを作る
   (b) すべてのクラスタの組の距離を計算する
2. 反復
   (a) もっとも距離の近いクラスタの組を併合する
   (b) 併合によって出来たクラスタと他のクラスタの距離を計算する
3. 停止条件の検査クラスタが一つになるまで反復操作を繰り返す

この中で 2. の反復の (b) において、2 つのクラスタ c_i と c_j を併合したクラスタ c_{ij} とクラスタ c_k との距離の計算の決め方によって、階層的クラスタリングはさまざまなアルゴリズムに分かれる。

階層的クラスタリングで用いられるクラスタ間の距離の式は以下のようなものがある。

- 単連結法 (single linkage method)

$$D(c_i, c_j) = \min_{x_i \in c_i, x_j \in c_j} D(x_i, x_j)$$

- 完全連結法 (complete linkage method)

$$D(c_i, c_j) = \max_{x_i \in c_i, x_j \in c_j} D(x_i, x_j)$$

- ウォード法 (Ward's method)

$$\begin{aligned} D(c_i, c_j) &= E(c_i \cap c_j) - E(c_i) - E(c_j) \\ E(c_i) &= \sum_{x \in c_i} (D(x, c_i))^2 \end{aligned}$$

単連結法は２つのクラスタの一番近い要素間の距離によってその２つのクラスタの距離を近似し、完全連結法は２つのクラスタの一番遠い要素間の距離で２つのクラスタの距離を近似する。ウォード法はクラスタの各値からそのセントロイドまでの距離の二乗の総和を最小化する。

3.3.2 非階層的クラスタリング

非階層クラスタリングは階層クラスタリングと異なり、クラスタリング対象となる文章集合が、幾つかの項目のクラスタにわかれた平坦な構造が結果として得られる手法である [図 3.5]。

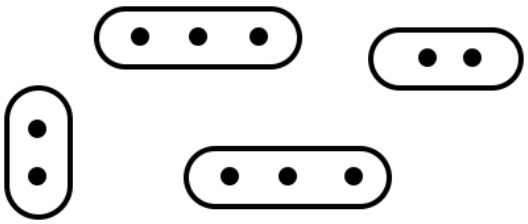

図 3.5: 非階層クラスタリング

大規模なデータを扱う場合、階層的な方法（単連結法、完全連結法など）の実行は難しい。なぜなら階層的な方法の場合、N 件の文書の各ペア（組）の類似度を求めなければならず、その計算量は、少なくとも、$O(N^2)$ になる。さらに、クラスタを階層的に構成するたあに、類似度データから適切な文書の組の類似度を探索するのにも一定の計算が必要となる。そのため情報検索の分野では早い時期から非階層的な方法の適用が探究されてきている。

非階層的クラスタリングの代表的な手法としては K-means が知られている。K-means はクラスターの数をたとえば K 個とあらかじめ固定しておき、以下のようなやり方でクラスタリングを行う。K-means ではそれぞれのクラスタに対応した、クラスタの中心、すなわちセントロイドを K 個の決める。それぞれのセントロイドは任意のものを選べるが、選択は結果に大きく影響を与えるため、なるべくそれぞれ離れたものが選択される。

次に、クラスタするすべての対象を最も近いセントロイドに割り当てる。すべて割り当て終わると、前段階の結果として得られたクラスタごとにセントロイドを計算し直し、新たに K 個のセントロイドを得る。また同様に、すべての文書についてK 個の新たなセントロイドのなかから最も近いセントロイドを割り当てる。以下、セントロイドが動かなくなるまで計算を続ける。

以上のアルゴリズムによって、次の目的関数が最小化される。

$$J = \sum_{i=1}^{k} \sum_{j=1}^{n} \| x_i^{(} j) - c_j \|$$

ここで、x_i^j はクラスタリング対象の文書 c_j はクラスタのセントロイドであり、目的関数はそれぞれの文書と、文書が属しているクラスタのセントロイドとの距離の総和である。

Kmeans のアルゴリズムをまとめると、下記のようになる。

1. K 個の初期セントロイドを設定する。
2. すべてのデータと K 個のクラスタのセントロイドとの距離を計算し、最も近いクラスタに割り当てる。
3. 新たに形成されたクラスタの中心を計算する。
4. 新たなセントロイドが前のセントロイドと異なっている場合 2. に戻る。そうでなければ収束しているとして、終了する。

クラスターの中心として平均値（mean）を用いることから、K 個の mean ということで、K-means 法と呼ばれている。

3.3.3 COP-KMeans

COP-KMeans は Wagstaff らが提案した、K-means にユーザの背景知識を利用できるよう拡張することで性能を向上させている手法の一つである [31]。COP-KMeans ではユーザの背景知識を、インスタンスレベル、すなわち文書レベルでの制約の集合として表現する。そして制約に違反しないようにクラスタリングをすることで、ユーザの背景知識をクラスタリング結果に反映する。

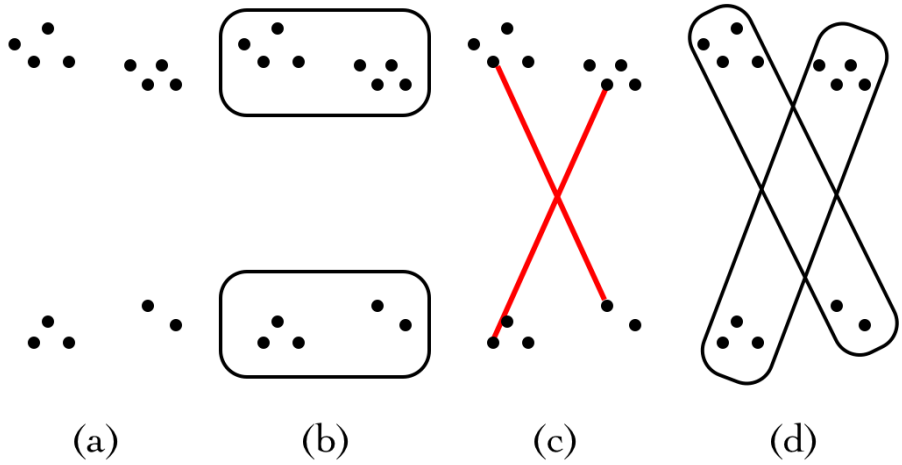

図 3.6: COP-Kmenas

たとえば、図 3.6 のような二次元空間に分布する文書集合があったとする。この文書集合を 2 つにクラスタリングしようとするとき、K-Means を用いると、(b) のような結果が得られる。しかし、そこでユーザが背景知識に基づいて (c) の赤い線のような制約を与えたとすると、(d) のような制約すなわち背景知識なしでは得られない結果がもとまる。

制約

COP-KMeans では、背景知識を文書レベルの制約として、 2 つの文書が同じグループに入るべきか、あるいは入らないべきかという形に変換する。よって一対の文書に対して次の 2 つの制約が考えられる。

Must-Link 2 つの文書は同じクラスターに含まれなければならない

Cannot-Link 2 つの文書は同じクラスターに含まれてはならない

must-link 制約は推移的に作用する。そのため、ユーザによって与えられた must-link 制約から結果として閉じた制約の集合が得られる。

たとえば A と B と C と D という文章が与えられ、A と B、B と C が must-link であるとすると、A と C もまた同様に must-link であるという制約が得られる。またこの場合、A と D が cannot-link であったとすると、B と D、C と D もまた cannnot-link になる。

得られた制約の集合は、次に述べるクラスタリングアルゴリズムで用いられる。

制約付クラスタリングアルゴリズム

COP-Kmeans のアルゴリズムは表 3.1 である。COP-Kmeans は文書集合 D、must-link 制約集合 $Con_{=}$、cannot-link 制約集合を入力として、文書集合 D の文書によって構成されるクラスタ集合を返す。得られたクラスタ集合は制約を満たしている。

K-means からの主な変更点は、それぞれの文書を最も近いクラスタに割り当てる際に、制約が満たされるように割り当てる点である。文書 d をクラスタに割り当てるとき、通常であれば最も近いクラスタ C_j に割り当てる。制約が破られていない場合はそれで問題ない。しかし、d と同じクラスタになければならない (must-link) 文書 $d_{=}$ があり、それがすでに C_j とは別のクラスタに割り当てられている場合。あるいは、d と同じくラスタにあってはならない (cannot-link) 文書 $d_{\neq}$ があり、すでに C_j に割り当てられている場合、d を C_j に割り当てることは出来ない。

d について制約を満たすクラスタが存在しないとき、アルゴリズムはからの集合 を返して終了する。

表 3.1: **COP-KMeans** のアルゴリズム

COP-Kmeans(文書集合 D, must-link $Con_{=} \subseteq D \times D$, cannnot-link $Con_{\neq} \subseteq D \times D$)

1. $C_1, ..., C_k$ を初期クラスタとする。
2. D 中のすべての文書 d を、それぞれ最も近いクラスタ C_j にわりあてる。ただし、C_j は VIOLATE-CONSTRAINTS が $false$ であるものとする。そのような C_j がないときは を返して終了する
3. それぞれクラスタ C_j について、クラスタを構成する全文書 d_j の平均をとることでセントロイドを更新する
4. 2 から 3 を収束するまで繰り返す。
5. $\{C_1, ..., C_k\}$ を返す

VIOLATE-CONSTRAINTS(文書 d, クラスタ C, must-link $Con_{=} \subseteq D \times D$, cannnot-link $Con_{\neq} \subseteq D \times D$)

1. $(d, d_{=}) \in Con_{=}$ について、もし $d_{=} \notin C$ であるとき $true$
2. $(d, d_{\neq}) \in Con_{\neq}$ について、もし $d_{\neq} \in C$ であるとき $true$
3. 1,2 以外の場合 $false$

3.3.4 制約付階層クラスタリング

制約付階層クラスタリングでは、COP-KMeansで用いられているmust-link、cannot-linkといった文書レベルでの制約を、階層的クラスタリングに用いて精度向上を行っている手法について述べる。また文書レベルの制約を、空間レベルの制約に変換して制約付クラスタリングを行う、Constrained Complete Linkageについても述べる。

must-link、cannot-link制約を用いた階層的クラスタリング

2つの文書が同じクラスタにあるべきである、あるいはないべきであるということを表す文書レベルの制約、must-link、cannot-link制約を階層的クラスタリングで用いるために、Dvidsonらは、表3.2のようなアルゴリズムを提案している[7]。

表3.2: 制約付階層クラスタリング

ConstrainedAgglomerative(文章集合 D, must-link ML,cannot-link CL)

1. ML制約から推移的に得られる閉じた文書の集合 $M_1, ..., M_r$ を計算する
2. もし2つの点 $\{x, y\}$ がMLとCLの両方に含まれている場合、{}を出力して終了する
3. $D_1 = D - (\sum_{i=1}^{r} M_i)$、$k_{max} = r + |D_1|$ とする
4. $M_1, ..., M_r$、及び D_1 に含まれる要素が一つからなるクラスタから合計 k_{max} 個の初期クラスタを構築する。
5. (a) 類似度関数に従ってCLを満たすもっとも近いクラスタペア C_l、C_m を選択する
   (b) クラスタ C_l を C_m に併合し、C_l を削除する
   (c) (a),(b)を併合できるクラスタがなくなるまで繰り返す。

アルゴリズムについて例を挙げて解説する。

クラスタリングしたい文書としてA,B,C,D,E,Fという6つの文書があったとする。また、制約条件としてmust-link制約の集合 $ML = (A, B), (B, C), (D, E)$ が与えられたとする。このとき、MLの(A,B),(B,C)という制約から、A,B,Cという三つの

文書が一つのクラスタにならなければならないという制約が得られる。また、同様に D,E も一つのクラスタにならなければならない。これは表 3.3 のアルゴリズムで計算する。

この場合、ML 制約から推移的に得られる閉じた文書の集合はそれぞれ $M_1 = A, B, C$、$M_2 = D, E$ となる。もしこのとき cannot-link として CL=(A,C) が与えられたとすると、A,C はともに M_1 に含まれるため、制約を満たす回は存在しなくなり 2. においてアルゴリズムは終了する。

must-link と cannot-link との矛盾がない場合、3. において D_1 が計算される。D から M_i に含まれる文書を除いたものが D_1 であるため、$D_1 = F$ となる。また $k_{max} = 2 + 1 = 3$ となる。k_{max} はクラスタの最大数である。

次に 4. で階層的クラスタリングにおける初期クラスタを計算する。初期クラスタは M_i 及び M_i に含まれない文書の集合 D_1 の要素が一つずつ含まれるクラスタであるため、例の場合 $M_1 = \{A, B, C\}, M_2 = \{D, E\}, D_{1_1} = \{F\}$ の三つのクラスタになる。

最後に 4. で計算した初期クラスタを使って、cannot-link を破らないように階層的クラスタリングを行う。

まとめると、must-link を満たすクラスタを作り、それを初期クラスタにする。そして cannot-link を破らないようにクラスタリングをすることで、制約を満たしたクラスタリング結果を得ている。

表 3.3: 閉じた must-link 制約の集合を求める手法

1. 各ノードがそれぞれ、個々の must-link 制約に対応しているような無向グラフ G を構築する
2. ２つのノードに対応する must-link 制約に含まれる文書に同じものがある場合、その２つのノードにエッジを張る
3. G において経路のある２つのノードの集合が、閉じた must-link 制約の集合

Constrained Complete Link

ここまで述べた制約付クラスタリングの手法はすべて、与えられた must-link、cannot-link という制約を、制約に明示的に指定されているインスタンスすなわち文書にのみ適用してクラスタリングを行っている。

しかし、ベクトル空間モデルでは文書が空間上に位置していることを考えると、制約を使ってベクトル空間における距離を計算し直すことで、制約に明示的に示されていない文書についても制約によって与えられる情報を用いることが出来る。

Klein らが提唱している Constrained Complete Link では、上記の考えをもちいて文書レベルの制約を、空間レベルに変換している [16]。Constrained Complete Link の概要は次のようになる。

まず、クラスタリング対象の文章集合に対して、図 3.3 のような距離行列を計算する。次に与えられた制約条件を反映するような、新しい距離行列を計算し直す。最後に新しい距離行列を使って、階層的クラスタリングを行う。

距離行列の変換には、制約によって同じクラスタに属さなければならない文書同士は近い距離に、違うクラスタに属さなかればならない文書同士は遠く離れるような結果が得られるように行う。また、次の何も制約のない文書についても次の 2 の条件が満たされるような変換を行う。

- 2 つの文書 d_i, d_j が近い場合、d_i に近い文書は d_j とも近い距離にある
- 2 つの文書 d_i, d_j が遠い場合、d_i に近い文書は d_j とは遠い距離にある

距離行列の変換は、制約の適用と伝搬、という 2 つの段階に分けて行われる。また、must-link と cannot-link はそれぞれ別に行われる。

must-link の適用と伝搬は次のように行う。CCL ではまず距離行列を文書をノードとしたときの完全グラフのエッジの長さと見る。そこで制約をかけられているペア間の重みをゼロとすることで must-link の適用が行われる。制約の適用を行うことで、直接距離を計算するのではなく、must-link の適用によって長さがゼロになったエッジに迂回して距離を計算することで、もとの距離より近くなるような文書の組が出てくる。must-link 制約を適用した距離行列に対して、すべてのノード間でもっとも短い距離を計算する all-pairs-shortest-path-length アルゴリズムで、迂回した方が近くなる文書組間の迂回距離をすべて計算する。

結果として得られる新たな完全グラフのエッジの長さを距離行列と考えることで、mustlink 制約を伝搬した新しい距離行列が得られる。

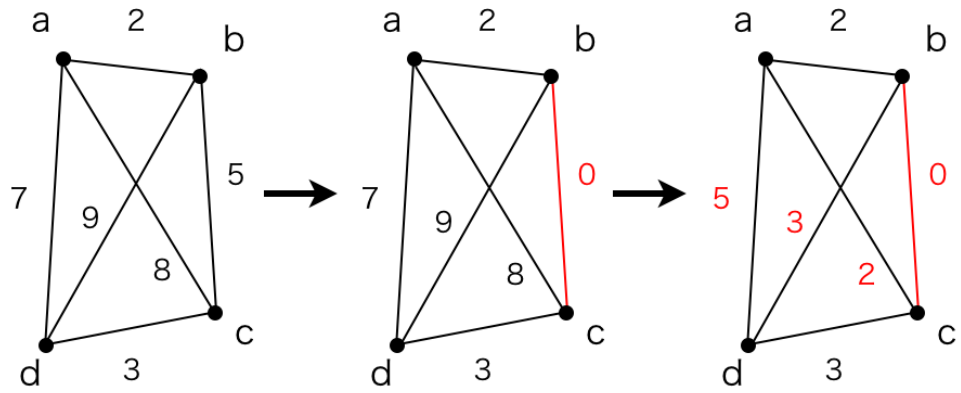

図 3.7: Constrained Complete Link

例を挙げて説明する。A,B,C,D という文書があり、グラフで表現したとき図**??**のようになるとする。もしここで B と C が must-link だったとすると、まず制約の適用として B,C 間のエッジの値が 0 になる。その結果 A,C 間、A,D 間、B,D 間の距離について B,C のエッジを経由して距離を計算した方が近くなる。そこで新しい距離を計算しなおすことで、制約を考慮した距離が得られ、制約の伝搬が行われたことになる。

ここでは、all-pair-shortest-path-length アルゴリズムとして、Floyd-Warshal のアルゴリズムを変形したものを用いる。

cannot-link の適用と伝搬は以下のようになる。まず事前に must-link の適用と伝搬を行って変形が終わっている距離行列を用意する。この距離行列における、cannot-link を与えられた文書ペア間の距離をすべて十分に大きな値、ここでは距離行列の要素の最大値より 1 だけ大きな値である、$max_{i,j}D_{i,j}$ とする。

cannot-link の適用はこれで完了する。

cannot-link の伝搬を must-link と同様に距離行列に対して適用しようとすると計算が困難な問題になる。そこで cannot-link の伝搬はクラスタリングアルゴリズムの実行時、クラスタの併合時に自動的に行われる。

クラスタリングには、完全連結法のアルゴリズムを用いる。階層的クラスタリングでは、最も近い距離にあるクラスタがはじめに併合され、離れているクラスタは最後に併合される。must-link が与えられたペア間の距離を 0、cannot-link が

与えられたペア間の距離を $max_{i,j}D_{i,j}$ とすることで、制約を階層的クラスタリングに適用することが出来ている。また、完全連結法では、クラスタ間の距離は、クラスタを構成する要素のうち、もっとも遠いもの同士の距離を用いる。そのため、違うクラスタに入らなければならない文書が含まれるクラスタ同士は、併合されない。よって、２つの文書 d_i, d_j が違うクラスタに含まれる場合、d_i に近い文書は d_j とは違うクラスタとなり、制約の伝搬の条件が満たされる。

Constrained Complete Link における制約の適用と伝搬アルゴリズムの擬似コードは図 3.8 階層的クラスタリングの擬似コードは図 3.9 となる。

constrainProximities(Matrix D, Constraints C)
  imposeMustLinks(D,C)
  propagateMustLinks(D,C)
  imposeCannotLinks(D,C)
  propagateCannotLinks(D,C)

imposeMustLinks(Matrix D, Constraints C)
  for $(i, j) \in C_{\text{must}}$
    $D_{ij}, D_{ji} = 0$

imposeCannotLinks(Matrix D, Constraints C)
  for $(i, j) \in C_{\text{cannot}}$ and $(j, k) \in C_{\text{must}}$
    $D_{ik}, D_{jk} = \infty$

propagateMustLinks(Matrix D, Constraints C)
  D = fastAllPairShortestPaths(D,C)
  for (i, j) s.t. $D_{ij} = 0$
    $C_{\text{must}} = C_{\text{must}} \cup \{(i, j)\}$

propagateCannotLinks(Matrix D, Constraints C)
  (done implicitly by completeLink)

fastAllPairsShortestPaths(Matrix D, Constraints C)
  % find valid intermediates
  $I = i : \exists j \neq i, (i, j) \in C_{\text{must}}$
  % modified Floyd-Warshall
  for $k \in I$, for $i \in \{1 : n\}$, for $j \in \{1 : n\}$
    $D_{ij} = \min\{D_{ij}, D_{ik} + D_{kj}\}$

図 3.8: 制約の適用と伝搬

constrainedCL(Matrix D**, Constraints** C**)**
  constrainProximities(D, C)
  completeLink(D)

completeLink(Matrix D**)**
  $Clusters = \{c_i$ for each point $i\}$
  $Linkage$ starts empty
  distances $\delta(c_i, c_j) = D_{ij}$
  while $|Clusters| > 1$
    choose closest $(c_1, c_2) = \arg\min_{c_1, c_2 \in Clusters} \delta(c_1, c_2)$
    add (c_1, c_2) to $Linkage$
    merge c_1 and c_2 into c_{new} in $Clusters$
    for $c_i \in Clusters$
      $\delta(c_i, c_{new}) = \max\{\delta(c_i, c_1), \delta(c_i, c_2)\}$

図 3.9: 階層的クラスタリング

3.4 制約の付け方

前節までで述べたように、制約付クラスタリングをもちいる場合、ユーザの背景知識を制約、この場合は must-link 制約と cannot-link 制約の２つで表現しなければならない。

提案手法の大まかな流れは次のようなものである。まずユーザが検索クエリを入力し、得られた初期検索の結果にたいして、ユーザが適合不適合の判定をおこなう。そして、適合不適合の判定情報を基に制約付クラスタリングが行われる。すなわち検索結果の文書ランキング上位に対して、適合文書、不適合文書、判定不能、の三つのラベルがあたえられ、その情報を基に制約付クラスタリングを行うこととなる。そのため、この文書に張られたラベルから、文書間の制約を生成しなければならない。

提案手法では、検索結果の文書ランキングに対して、次のような仮定を置いている。文書ランキングは複数のトピックの文書集合の集合であり、ユーザが求める検索結果は、その中の一つのトピックである。また、不適合文書は複数のトピックの文書集合から構成されうる。

以上のような仮定から、制約付クラスタリングの結果としては適合文書は一つのクラスタを形成し、不適合文書は複数のクラスタを形成されることが望ましい。そこで、提案手法では文書間の制約を次のように設定する。

- 適合であるとラベルづけされた文書間にはすべて must-link 制約をつける
- 適合であるとラベル付けされた文書と、不適合であるとラベル付けされた文書間にはすべて cannot-link 制約をつける

判定不能のラベルについてはこの場合有用な情報を持たないので、提案手法ではもちいない。

ただし、初期検索の結果として得られる検索ランキングの上位に、適合文書と不適合文書が必ずしも両方存在するとは限らない。また、仮にあったとしても、ユーザが適合と不適合のどちらか片方だけしか入力として与えない場合もあり得る。そのため、ユーザからの適合不適合判定情報の入力には次の三つの場合が考えられる。

1. 適合判定、不適合判定がともに入力される

2. 適合判定のみが入力される

3. 不適合判定のみが入力される

それぞれの場合、文書間の制約は図 3.10 のように与えられる。

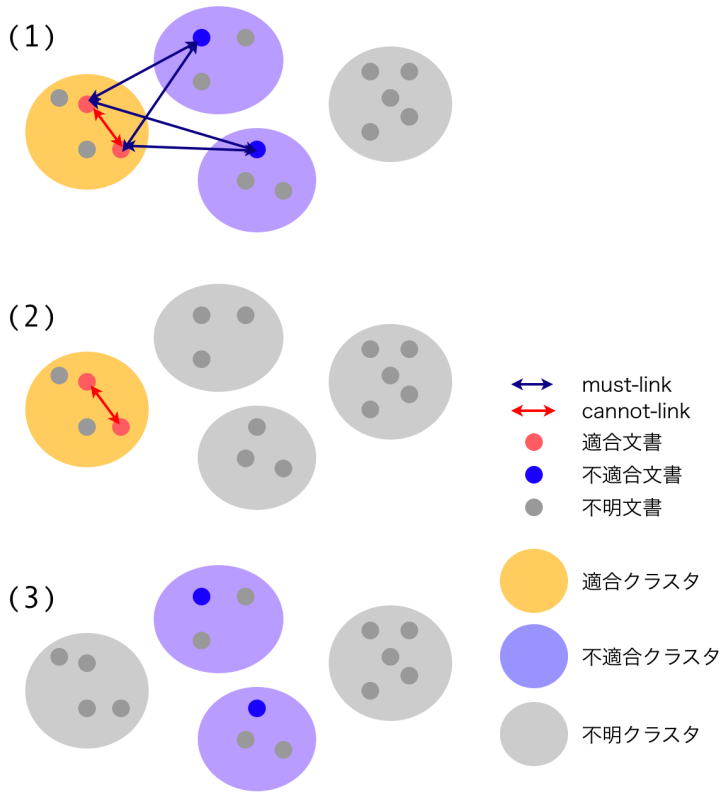

図 3.10: 制約の付け方

まず 1. の場合すなわち両方の情報が使える場合は、適合文書同士を must-link で結び、適合文書と不適合文書が cannot-link でむすばれる。2. の場合、不適合文書がわからないので、適合文書同士が must-link で結ばれるのみだる。また、3. の場合は適合文書がないため、must-link、cannot-link があたえられれない。なぜなら両制約とも、ペアの片方が適合文書だからである。そのため制約を与えず、通常の文書クラスタリングを行う。不適合判定情報のユーザ入力は、クラスタリング結果に対して、適合不適合クラスタのラベル付けを行うときに用いられる。

3.5 フィードバックの実現

制約付クラスタリングした結果、文書のクラスタが複数得られる。本節では得られた文書クラスタを用いて、新しい検索結果をどう得ているかについて述べる。

得られた文書クラスタに、適合文書クラスタ、不適合文書クラスタ、その他クラスタのラベルを付与することで、後で適合性フィードバックに用いる擬似適合文書集合、疑似不適合文書集合とする。クラスタのラベル付けは次のようにする。

文書クラスタのうち、ユーザによって適合と判定された文書が含まれているクラスタは適合文書クラスタ。不適合と判定された文書が含まれているクラスタは不適合文書クラスタとして、どちらも含まないクラスタはその他クラスタとして、後の処理では用いない。前節で述べた制約の付け方から、適合文書クラスタは一つ、不適合文書クラスタは複数得られる。

関連研究の章で述べたように、Rochio の手法に代表されるベクトル空間モデルでの適合性フィードバックの手法は、基本的に以下の式で表される。

$$Q_{n+1} = Q_n + \alpha \sum_{j=1}^{N^+} D_j^+ - \beta \sum_{j=1}^{N^-} D_j^-$$

しかしこの式では、適合性フィードバックに用いられる文書は、適合と不適合という２つの文書集合しかないことが前提となっている。

しかし提案手法では、ラベル付けの結果として、複数の不適合文書クラスタが得られる。そのため、不適合クラスタをまとめて一つの文書集合としてフィードバックを行ってしまうと、クラスタリング結果の情報が失われてしまうと考えられる。そこで、本研究では、以下のように適合性フィードバックの式を変形することで、複数の不適合文書クラスタがあることを考慮した適合性フィードバックを行っている。

$$Q_{n+1} = Q_n + \frac{\alpha}{N^+} \sum_{j=1}^{N^+} D_j^+ - \beta \sum_{i=1}^{N_{C-}} \frac{1}{N_i^-} \sum_{j=1}^{N_i^-} D_{ij}^-$$

ここで、N^+, D_j^+ は適合文書クラスタの文書数と、適合文書、$N_{c\_}$ は不適合文書クラスタの数、N_i^- は不適合文書クラスタの文書数、D_{ij}^- は i 番目の不適合文書クラスタに含まれる不適合文書である。また α, β, γ はそれぞれ、前の検索クエリ、適合文書、不適合文書をそれぞれどのていど重要視するかを表すパラメータである。

上記式で複数の不適合文書クラスタを考慮した新しい検索クエリが生成される。新しい検索クエリを使って、初期検索と同様に検索をもう一度行うことで、あたらしい検索結果である文章ランキングが得られ、ユーザに提示される。新しい検索結果がユーザのニーズを満たさない場合、再びユーザが適合性判定を行うことで、もう一度適合性フィードバックが行われる。

第4章　実験・評価

本章では、提案した適合性フィードバック手法の実装と実世界データを用いた実験について詳細を述べる。また、実験結果について、考察を行う。

4.1 実装・実験の概要

本節では、提案手法の実装の概要と、実験について述べる。

4.1.1 前処理

提案手法を実現するため、実験では以下のような処理を行う。まず、多量の文書からなる文書コレクションを準備する。前処理として文書コレクションに対して以下のようなプロセスの処理をすることで逆引きインデックスを作成する。

1. 各文書を単語に切り分ける
2. 単語を正規化する
3. 正規化された単語からストップワードなどを取り除く
4. 3. で残った単語から文書-単語インデックスを作成する
5. 文書-単語インデックスから逆引きインデックスを作成する
6. 逆引きインデックスの各単語を重み付けする

英語の場合、文書を単語に切り分けるのは、空白を区切りとして分割を行うだけで良い。単語の正規化とは、同じ単語が活用されているため表記上異なってしまっている場合、それらの違いを吸収して同じ表記にすることである。ここではstemmer をもちいて単語の語幹への変換を行う。

日本語の場合は、文書に対して形態素解析ソフトをかけ、単語単位に分割する。また、正規化としての原形への変換も同時に行われる。

ストップワードとは、検索をする上で、あまり役に立たない、むしろ性能を落としてしまう一般的すぎる単語のことである。たとえば英語で言うと、”about”や”of”といった単語である。ストップワードをあらかじめ除くことで、検索性能の向上を図っている。

文書-単語インデックスは、図 3.3 の縦ベクトルに対応し、各文書にどういった単語がどれだけの頻度で現れているかを示している。そして文書-単語インデックスから逆引きインデックスである単語-文書インデックスを作成する。これは、図 3.3 の横ベクトルに対応し、それぞれの単語をどの文書がどれだけ含んでいるかを

示している。本研究では逆引きインデックスは実装を容易にするため、ハッシュと配列を用いたシンプルな実装を行っている。

最後に逆引きインデックスに提案手法で述べた単語の重み付けをすることで前処理が完了する。検索時には、検索クエリに含まれる単語がどの文書に含まれているかを逆引きインデックスから求めることで、効率よく検索結果を求める。

4.1.2 実験

実験には用意した文章コレクションに対応した、質問セットと正解データを用いた。質問セットは検索クエリと、検索クエリについての説明であり、正解データは各検索クエリで検索したとき、文書コレクションの各文書が適合か不適合か記されているものである。実験は次のような流れで行った。

1. 質問セットの検索クエリを用いて初期検索
2. 初期検索の上位 N 件に対して正解データを用いて適合不適合のラベルを付ける
3. 適合不適合のラベルから制約を生成する
4. 制約を用いて初期検索の上位 M 件をクラスタリングす
5. 結果として得られたクラスタに適合不適合のラベルを付ける
6. 適合、不適合クラスタを使って新しいクエリを生成する
7. 新しいクエリで検索
8. 正解データを用いて性能評価

検索には逆引きインデックスを用いる。検索クエリに含まれる単語を持つ文書のスコアを逆引きインデックスからも止めることでベクトル空間モデルの計算を実現している。ただし、検索クエリを単語ベクトルにするさい、ここでは単に単語の頻度をそのまま用いている。

2. については実験では、人手で上位 N 件に対して適合性判定を行う代わりに、質問セットの正解データを用いて適合性判定を行っている。

制約の生成と、クラスタリングは、提案手法に述べたように行う。ただし $N < M$ である。またクラスタリング時には、文書の単語ベクトルが必要となるが、これ

にはインデックス作成時に用いた文書-単語インデックスをつかう。この文書-単語インデックスの重みは単純な単語頻度であるため、提案手法で述べた重み付けも同時に行う。

クラスタのラベル付けも提案手法で述べたように、それぞれ適合、不適合文書が含まれているかで判断する。

新しいクエリを用いた再検索は、初期検索と同様に行う。

提案手法の実験の比較対象となるベースライン手法としては、関連研究で述べた、Rocchio の手法を用いた。

提案手法では、単語の重み付け手法、クラスタリング手法、距離関数など様々なパラメータが存在する。そのため、予備実験として、初期検索の検索性能を評価することで、単語の重み付け手法の選択を行っている。また、制約付クラスタリングの性能表かを行うことで、いくつか用いる要素手法の選択を行っている。

4.2 実験環境の詳細

本節では、実験に用いた計算機、プログラムなどについて解説する。

4.2.1 計算機環境

実験には計算機環境として、図 4.1 で構成される計算機 16 台と図 4.2 の RAID 装置からなる PC クラスタを用いた。

Xeon 3.2DGHz/1MB/FSB800 x 2
8 GB memory PC3200 DDR2 ECC registered (1 GB x 8 , slot=8)
400GB 7200rpm PATA disk x 1 (disk slot =3)
E7520 chip set
Gbit ethernet x 2
DVD-ROM (x8 DVD , x24 CD-ROM)
2 mode FDD
ATI RageXL 8MB video
PCI Express slot x 2
ATA133R-2D50 smart IDE cable x 2
4PW-2D40 power cable x 2
OS :Fedra core 7 x86-64

図 4.1: 計算機の構成 1

250GB disk x 16
RAID5 設定時容量 = 3750GB
RAID5 + hot stanby 設定時容量 = 3500GB
5132MB memory
RAID0,1,3,5,6,JBOD サポート
2 Gbit Fibre(copper) for host x 2

図 4.2: RAID 装置の構成

4.2.2 利用したプログラム

実験には、以下のようなプログラムを用いた。

Ruby

提案手法実装のためのプログラミング言語としては、まつもとゆきひろにより開発されたオブジェクト指向スクリプト言語である Ruby のバージョン 1.8.6 を用いた。

Ruby はスクリプト言語であり、C 言語などに比べるとプログラミングの効率がよいが、一方でインタプリタ型であるため実行速度が遅いという不利な点がある。しかし、今回の実験では、リアルタイムの実験ではないこと、実行時のボトルネックはむしろ IO にあることから、開発の効率がよいと云われる Ruby を用いた。

Berkeley DB

テストデータの文書コレクションを記憶するデータベースには Berkeley-DB を用いた。Berkeley DB はオープンソースで公開されている、アプリケーション組み込み型のデータベースライブラリである。SQL のようなデータ操作言語を持たず、データベースへのアクセスは全てサブルーチン呼び出しによって行う。

Ruby ではバインディングである bdb を用いることで、ハッシュや配列と同様の容易さでプログラム中でデータベースへとアクセスできる。

Porter Stemmer

stemming とは単語の複数形や変化形などの類義語を検索できるようにするための技術であり、stemmer は語形変化や派生語による言葉のゆれを解消するために、単語から互換のみを抽出する処理をするプログラムである。本研究では stemmer として Poter Stemmer を用いた。

Poter Stemmer は一定の規則に従って単語の接尾辞を変形させることで、単語の変化を吸収している。たとえば、”ambiguous”と ”ambiguity ”という 2 つの単語がある場合、Poter Stemmer で Stemming を行うと、”ambigu”となり、同一単語の変化形であることがわかる、。

Poter Stemmer は辞書を用いないルールベースであるため、どんな新しい単語にも対応できるという利点がある。

```
これは形態素解析ソフトのテストです

これ	名詞, 代名詞, 一般,*,*,*, これ, コレ, コレ
は	助詞, 係助詞,*,*,*,*, は, ハ, ワ
形態素	名詞, 一般,*,*,*,*, 形態素, ケイタイソ, ケイタイソ
解析	名詞, サ変接続,*,*,*,*, 解析, カイセキ, カイセキ
ソフト	名詞, 一般,*,*,*,*, ソフト, ソフト, ソフト
の	助詞, 連体化,*,*,*,*, の, ノ, ノ
テスト	名詞, サ変接続,*,*,*,*, テスト, テスト, テスト
です	助動詞,*,*,*, 特殊・デス, 基本形, です, デス, デス
EOS
```

図 4.3: 和布蕪の出力結果

和布蕪 (mecab)

和布蕪は条件付き確率場をもちいたオープンソース形態素解析である。また、和布蕪は言語, 辞書, コーパスに依存しない設計となっている。また、形態素解析ソフトとして有名な ChaSen や KAKASI に比べ高速であるという特徴を持つ。本研究では、辞書として IPA 辞書を用いた。

和布蕪では形態素解析をすることで、日本語文章の分かち書きが得られる。また、それぞれの単語について、品詞や原形、読みの情報が得られる。たとえば「これは形態素解析ソフトのテストです」という入力を入れた場合、和布蕪は図 4.3 のような結果を返す。

4.3 評価手法

提案手法において評価を行うポイントは、大きく分けて２つに分けられる。一つは適合性フィードバックした結果として得られる文書ランキングであり。もう一つは、制約付クラスタリングの結果得られる適合クラスタ、不適合クラスタである。システム全体の性能表かは文書ランキングによってなされるが、手法の選択、パラメータの設定などにおいてクラスタの性能が用いられる。

4.3.1 検索結果の評価

情報検索システムの性能を評価する方法は様々な手法が提案されている。主な手法としては、精度と再現率という２つの指標がある。

精度とは、検索された適合文書数と検索された文書数の比率であり、検索結果にどれだけ適合した文書が含まれているかを示している。一方、再現率は、検索された適合文書数と、検索対象となる文書集合中の適合文書数の比率であり、どれだけ適合文章が漏れなく検索されたかを示している。

本研究では、検索性能の評価には、検索結果の精度を用いた。また、精度を算出するのにもちいる検索結果の文書は、文書ランキング上位N件までの見ている。すなわち、上位５件まで見たときの精度、上位十件まで見たときの精度……というように、精度を計算しグラフを得る。検索エンジンではユーザは検索ランキングの上位から順に見ていくため、この精度-ランキンググラフが高ければ高いほど評価が高くなると云うのは、直感にもあっている。

精度は次の式で表される。

$$P = \frac{w}{w + y}$$

ここで、wは検索された文書のうちの適合文書数、yは不適合文書数である。

システムの評価は、複数の検索質問に対して行われるため、個々の質問にたいする評価から全体の評価を得なければならない。ここではそのために精度のマクロ平均をとった。精度のマクロ平均は次の式で表される。

$$\overline{P} = \frac{1}{Q} \sum_{i=1}^{Q} \frac{w_i}{w_i + y_i}$$

ここで、w_iはi番目の検索質問で検索された文章のうちの適合文書数y_iは不適合文書数である。また、Qは全質問数である。

num_q	all	50
num_ret	all	47836
num_rel	all	16386
num_rel_ret	all	4593
map	all	0.1098
gm_ap	all	0.0342
R-prec	all	0.1860
bpref	all	0.1970
recip_rank	all	0.6391
ircl_prn.0.00	all	0.6666
	(中略)	
P5	all	0.4480
P10	all	0.4400
P15	all	0.4160
P20	all	0.4000
P30	all	0.3800
P100	all	0.2836
P200	all	0.2210
P500	all	0.1421
P1000	all	0.0918

図 4.4: TREC EVAL の出力

ただし、後述するように、本研究では質問の難易度を考慮し、検索質問を難しい質問と簡単な質問に分けて精度のマクロ平均をとる評価も行っている。

TREC EVAL

検索の性能評価で制度を計算するために、TREC EVAL というプログラムを用いた。TREC EVAL は TREC という検索タスクの性能を評価するために開発されたアメリカ国立標準技術研究所の基準に準拠したプログラムであり、検索に関する研究で使われている。

本研究で用いる、検索結果上位 N 件においての精度のマクロ平均も、図 4.4 のような出力によって得られる。ここで、P5 は上位 5 件までを見たときの精度、P10 以下も同様である。

4.3.2 クラスタリングの評価

クラスタリングの結果の評価は、検索結果とは異なり、適合度と再現率を同時に考慮したF尺度を用いる。これは、精度が高くなければ正しい適合性フィードバックが行われないが、一方で再現率が低い、つまり擬似的な適合判定情報があまり多くない場合、やはり適合性フィードバックの結果の改善が得られないと考えられるからである。

クラスタリングの結果得られた複数の文書クラスタから、一つの適合クラスタと複数の不適合クラスタを得る。これが評価対象となる。その他のクラスタは評価に用いない。適合クラスタはそのまま、不適合クラスタは合併して一つのクラスタしてしまう。クラスタの精度 P、再現率 R、及びF尺度 F を求める式は以下の式になる。

$$
\begin{aligned}
P &= \frac{w}{w+y} \\
R &= \frac{w}{w+x} \\
F &= \frac{2}{\frac{1}{P}+\frac{1}{R}}
\end{aligned}
$$

ただし、このとき、w は適合クラスタに含まれる適合文書数、y は不適合文書数、x は適合クラスタに含まれない適合文書数である。不適合クラスタの場合は、全て逆になる。

4.4 TREC

The Text REtrieval Conference(TREC) はアメリカ国立標準技術研究所 (NIST) とアメリカ国防総省内の研究部門である ARDA (Advanced Research and Development Activity) が主催している情報検索関連の研究分野に注目し開催されているワークショップであり、1992 年に TIPSTER Text プログラムの一部として始まったものが発展しながら継続してきている。

本節では、TREC の文書コレクションと検索タスクについて述べ、そのタスクについて行った実験結果、考察について述べる。

4.4.1 実験データについて

TIPSTER1

ここでは実験データとなる文書コレクションとして、TIPSTER1 を用いた。

TIPSTER1 は、 次のような文書から構成される文書コレクションであり、文書数は 287514、データサイズは 1251780 キロバイトである。

WSJ Wall Street Journal (1987, 1988, 1989),

AP AP Newswire (1989),

ZIFF articles from Computer Select disks (Ziff-Davis Publishing),

FR Federal Register (1989),

DOE short abstracts from Department of Energy(DOE) publications,

文書は次のように、SGML のような形式でファイルに記述されている。

<DOC>
< DOCNO > WSJ880406-O090 </DOCNO>
<HL> AT&T Unveils Services to Upgrade Phone Networks Under Global Plan </HL>
<AUTHOR> Janet Guyon <WSJ Staff> </AUTHOR>
<DA TEL INE> NEW YORK </DA TEL INE>
<TEXT>
American Telephone & Telegraph Co. introduced the first of a new generation of phone

services with broad implications for computer and communications equipment markets. A T& T said it is the first national long-distance carrier to announce prices for specific services under a world-wide standardization plan to upgrade phone networks. By announcing
commercial services under the plan, which the industry calls the Integrated Services Digital
Network, AT& T will influence evolving communications standards to its advantage, consultants said, just as International Business Machines Corp. has created de facto computer
standards favoring its products.
</TEXT>
</DOC>

TREC-1 ad hoc

質問セットには、TREC-1 ad hoc タスクを用いた。

ad hoc 検索とは、文書集合は検索質問に比べると比較的固定されていることから、検索質問を短期的、あるいは動的、文書集合を長期的、あるいは静的なものとしてとらえる情報検索のことである。

TREC-1 ad hoc タスクは全部で 50 の検索クエリから構成されており、TIPSTER1 の文書と同様に、次のような SGML のような形式で記述されている。また、本研究では初期クエリとして、<title> の項目を用いた。

<top>
<head> Tipster Topic Description
<num> Number: 066
<dom> Domain: Science and Technology
<title> Topic: Natural Language Processing
<desc> Description:
Document will identify a type of natural language processing technology which is being developed or marketed in the U.S.
<narr> Narrative:

A relevant document will identify a company or institution developing or marketing a natural language processing technology, identify the technology, and identify one or more features of the company's product.
<con> Concept<s>:
1. natural language processing
2. translation, language, dictionary, font
3. software applications
<fac> Factor<s>:
<nat> Nationality: U.S.
</fac>
<def> Definition<s>:
</top>

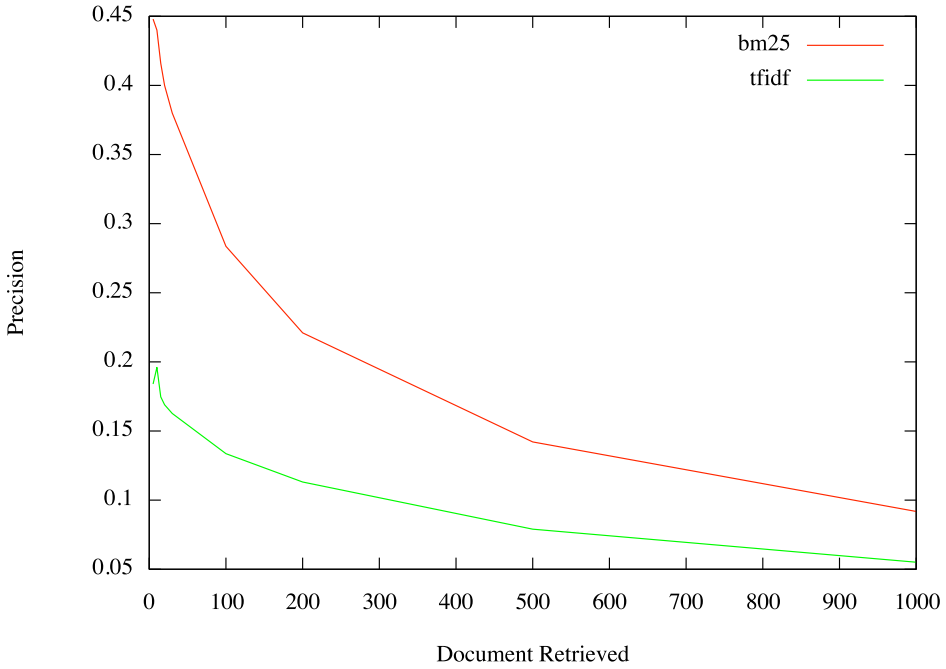

図 4.5: 単語の重み付け手法の比較

4.4.2 実験・結果

単語重み付け手法の選択

TREC1 の実験をするにあたり、予備実験として、まず検索用の逆引きインデックスにおいてどの単語の重み付けを行うべきかを実験した。本研究では TF*IDF と bm25 について比較を行った。また類似度の計算には、cos 尺度を用いている。

図 4.5 の結果から、以下の実験では逆引きインデックスにおける単語の重み付けとして、全て bm25 を用いている。

クラスタリング手法の選択

本小節では、クラスタリング手法の選択を行った。

本研究では、COP-KMeans、制約付階層クラスタリング、Constrained Complete Link(CCL) の 3 つの手法について調べた。また、制約付階層クラスタリングについては、クラスタ同士の類似度尺度について、単連結法、完全連結法、ウォード法の三つについて調べているため、合計 5 つの場合について調べている。

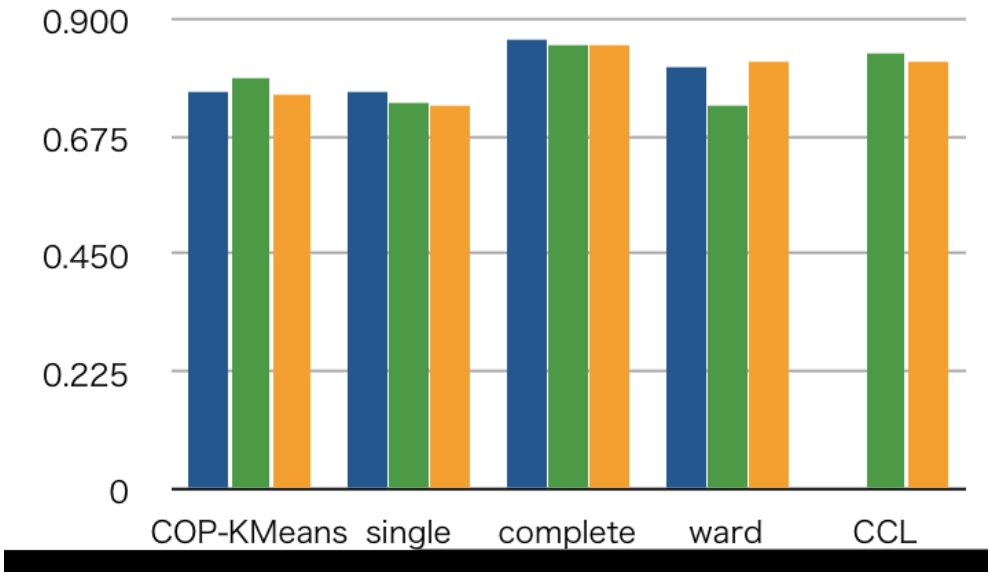

図 4.6: クラスタリング手法の比較

また、それぞれのクラスタリングについて、幾つかの文書間の類似度について調べている文書間の類似度として、コサイン尺度、ユークリッド距離、Jensen-Shannon エントロピーを用いた。CCL では、距離を用いなければならないためコサイン尺度は使えない。そのため、全 14 の場合について調べた。

クラスタリングに関するその他のパラメータは固定している。

図 4.6 の結果より本研究ではクラスタリング手法として、完全連結法の制約付階層クラスタリングを用いる。クラスタリング性能には主に手法の影響が大きいが類似尺度としては完全連結法でもっとも性能が良かったコサイン尺度を用いる。

クラスタリング時のパラメータの設定

本小節では、クラスタリングのパラメータについて調べた。本研究では、クラスタリングのパラメータとして、クラスタ数とクラスタ対象の文書のウィンドサイズがある。

文書のウィンドサイズとは、クラスタリング時に文書のどれだけの部分を単語ベクトルに変換するかであり、たとえばウィンドウサイズが 20 であれば、文書中の単語のうち、検索クエリに含まれる語の前後 20 語を単語ベクトルに変換する。

実験の結果、図 4.7 が得られれた。ここで凡例はクラスタ数である。図 4.7 からパラメータに対して敏感に変化することがわかる。ここでは以降の実験でもっと

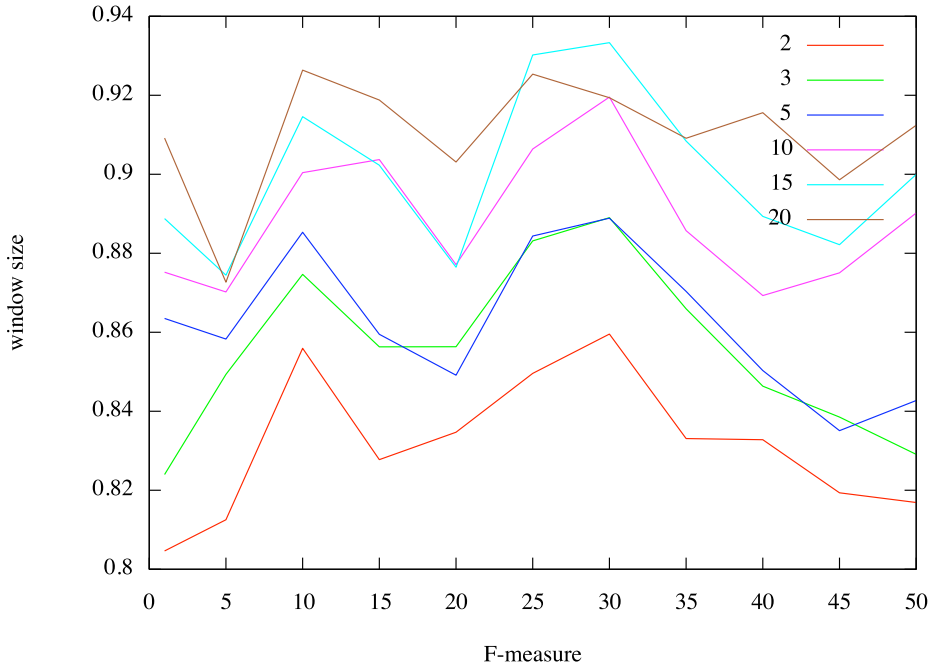

図 4.7: クラスタリングパラメータの決定

も性能の高いクラスタ数 15、ウィンドウサイズ 30 を用いる。

評価実験

初期検索、ベースラインと提案手法を比較したところ図 4.8 のような結果が得られる。ここで、initial_search は初期検索、baseline は関連研究で述べた Roccio の手法により得られる結果である。また、パラメータは、文献 [21] に従い、 に設定している。

図 4.8 より、提案手法では初期検索に比べて、フィードバックの結果大きな性能向上が得られていることがわかる。ただし、ベースラインからは向上しているものの、顕著な改善は見られない。

ここで、フィードバックのベースとなる初期検索の結果を考える。本論文ではここまで検索結果の評価にあたり、50 の質問セット全てを同様に取り扱い平均をとっている。しかし、質問セットには、良い検索結果が得られやすい質問と、良い結果を得るのが困難な質問が混じっていると考えられる。そこで、そういった簡単な質問と、難しい質問を考えるために、初期検索の結果上位 10 での精度がどのように分布しているかを調べると、図 4.9 のような分布が得られた。

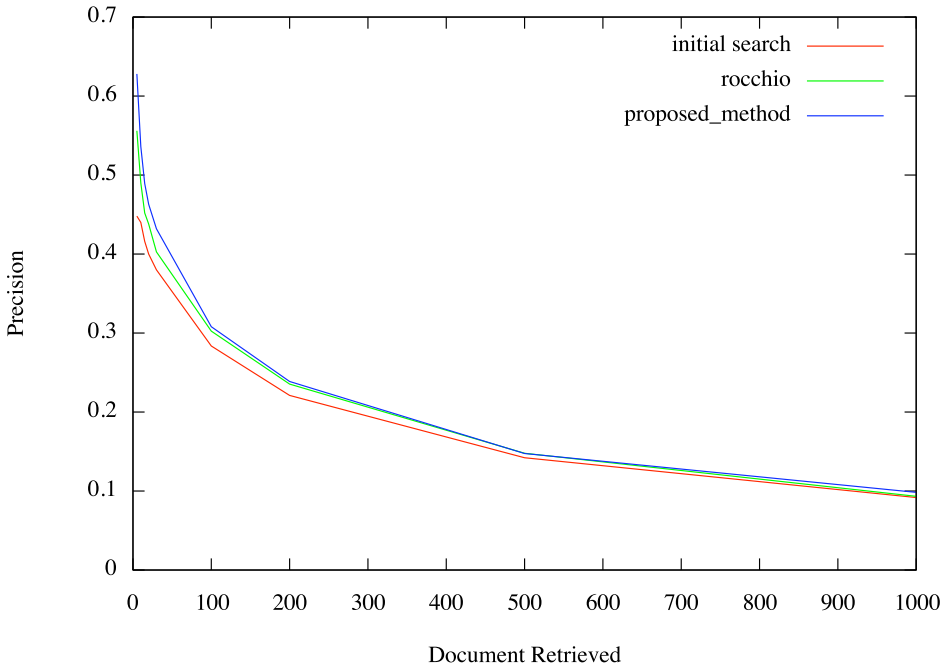

図 4.8: ベースラインとの比較

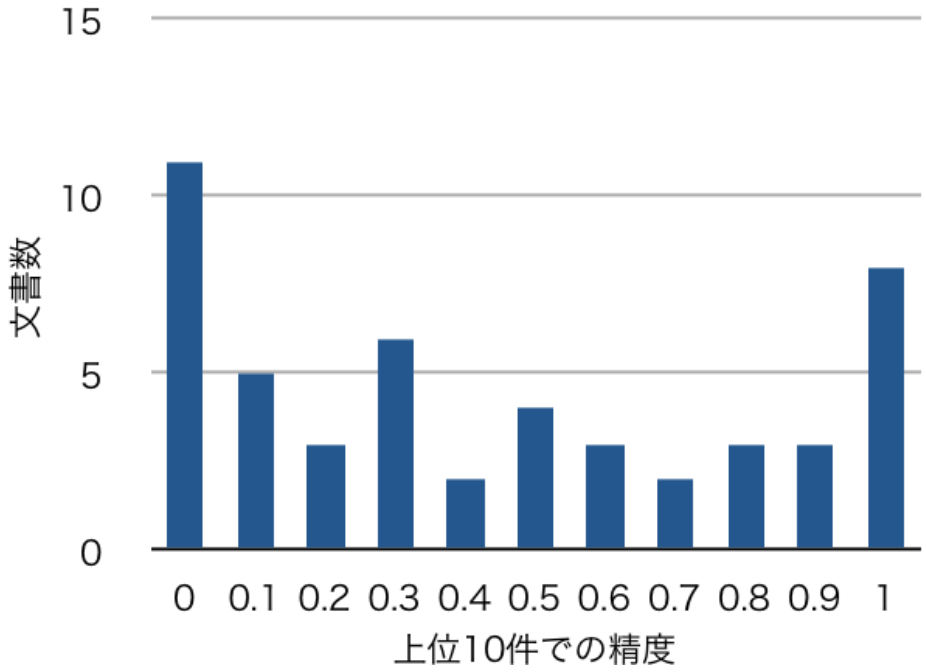

図 4.9: 上位 10 件での精度の分布

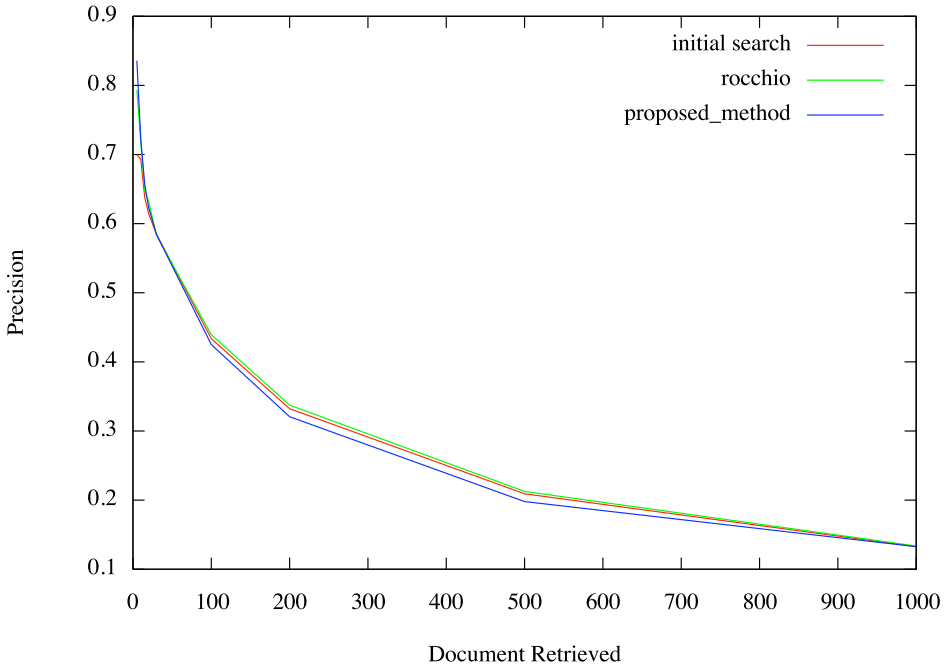

図 4.10: 簡単な質問に対する結果

この分布から、初期検索の結果上位 10 件の精度は質問によって大きく異なることがわかる。適合性フィードバックは検索結果の文書ランキングに適合不適合の判定を行う。そのため上位の精度にフィードバック結果は大きく影響を受けると考えられる。そこで、上位十件の精度が高いものと低いものを、それぞれ易しい問題、難しい問題と考えて、再度実験をおこなった。具体的には、上位十件での精度が 0.4 以上の質問と、0.3 以下の質問とに分けて実験を行った。これは質問がほぼ半分になるように分けている。その結果簡単な質問については、図 4.10、難しい質問については図 4.11 という結果が得られた。

簡単な質問については、提案手法はベースラインからの改善はほとんど見られなかった。しかし、ベースラインもまた、初期検索からの精度向上はほとんどない。一方、難しい質問については提案手法は、ベースラインを大きく上回る改善が見られた。

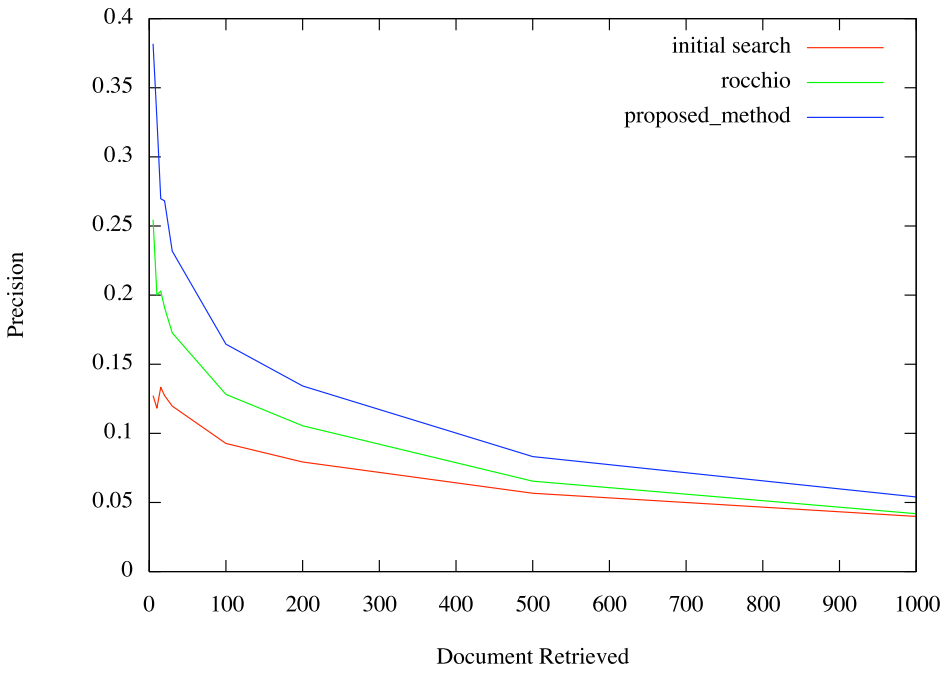

図 4.11: 難しい質問に対する結果

4.4.3 考察

本小節では、提案手法における要素技術の選択とパラメータの設定を行い、TREC1 ad hoc タスクの質問セットを用いて、ベースラインとの比較を行った。

実験結果から質問セット全体を見た場合、提案手法は初期検索からの改善は大きく見られたものの、ベースラインに比べてさほどの向上は得られなかった。しかし、質問セットを初期検索の結果によって、難しい質問と、簡単な質問とに分けた場合で大きく異なる結果が得られた。

簡単な質問での結果を見ると、提案手法はベースラインから性能の向上は見られるものの、改善の程度は小さい。しかし一方で初期検索とベースラインを比べたところ、やはり結果はあまり改善されていないことがわかる。よって、ベースライン、提案手法ともに適合性フィードバックの効果はあまり出ていないということがわかる。すなわち簡単な問題の場合、すでに初期検索の時点で十分な性能が出てしまっていると考えられる。また、実際にユーザが検索をする時も、初期検索の結果がよければ適合性フィードバックは必ずしも必要とされない場合が多いと思われる。

一方、難しい質問では、初期検索の結果がよくないため、実際にユーザが利用する場合でも検索結果の改善が必要となる。この場合、実験結果からもわかるように、提案手法では、ベースラインを大きく上回る検索結果の改善が見られた。難しい質問では初期検索の結果に改善する余地が大きいため、提案手法における適合文書と不適合文書を制約付クラスタリングで擬似的に増やしたことによる効果が大きく出ていると考えられる。難しい質問で大きな改善が得られているのに、全体で見たときあまり結果が改善されているように見えないのは、全体で見たとき全ての質問に対して単純な平均をとってしまっているため、良い性能が出る簡単な質問の結果に引きずられてしまっているからである。

よって提案手法は、初期検索の結果が良い場合はベースラインと同程度であるが、結果の改善がより必要となる初期検索の結果が悪い場合には、有効であるということがわかる。

4.5 NTCIR

NTCIR は正式名称「情報検索システム評価用テストコレクション構築プロジェクト」（NII-NACSIS Test Collection for IR Systems）という国立情報学研究所が 1998 年から行なっている TREC 形式の共同研究プロジェクトである。

本節では、NTCIR の文書コレクションと検索タスクについて述べ、そのタスクについて行った実験結果、考察について述べる。

4.5.1 実験データについて

ここでは、実験データとなる文書コレクションとして、NTCIR3 Web タスクの文書データを用いた。これは、Web から robot により収集した Web ページ であり、容量 100GB，総分少数は 2000 万程度の文書コレクションである。

robot の収集範囲は以下のようになっている。

- 対象サイト：.jp ドメインを中心とした http サーバ
- 対象ポート：すべて
- その他のページ：収集ページのリンク先のみ
- ファイルフォーマット：HTML，PlainText

用いたタスクは従来の学術文書や新聞記事を対象とした Ad Hoc 型検索に相当するサーベイ検索であり固定した文 書集合に対して，新たな検索課題で検索を行なう。ここで用いたタスクは全部で 40 の検索クエリから構成されており、次のような形式で記述されている。

<TOPIC>
<NUM>0041</NUM>
<TITLE CASE=”c” RELAT=”1-2”> 印象派, モネ, 美術館 *</TITLE>*
<DESC> 印象派に属する画家の絵画がどこで見られるかを探したい *</DESC>*
<NARR><BACK> 印象派とはモネの作品「印象・日の出」という題に由来するという。*</BACK><TERM>* 絵画で、印象主義をおしすすめた人々を印象派という。*</TERM><RELE>* 適合文書は印象派に属する画家のみではなくその作品（題名だけでも良い）も紹介されていて、どこの美術館でそれらの作品が見られるかとい

う情報を提供しているもの。*</RELE></NARR>*
*<CONC>*印象派、モネ, マネ, シスレー, ピサロ, 絵画, 美術館, ルイ＝ルロワ *</CONC>*
<RDOC>NW010642903, NW003795687, NW003948114</RDOC>
<USER> 大学院 *2* 年, 女性, 検索歴 *4* 年 *</USER>*
</TOPIC>

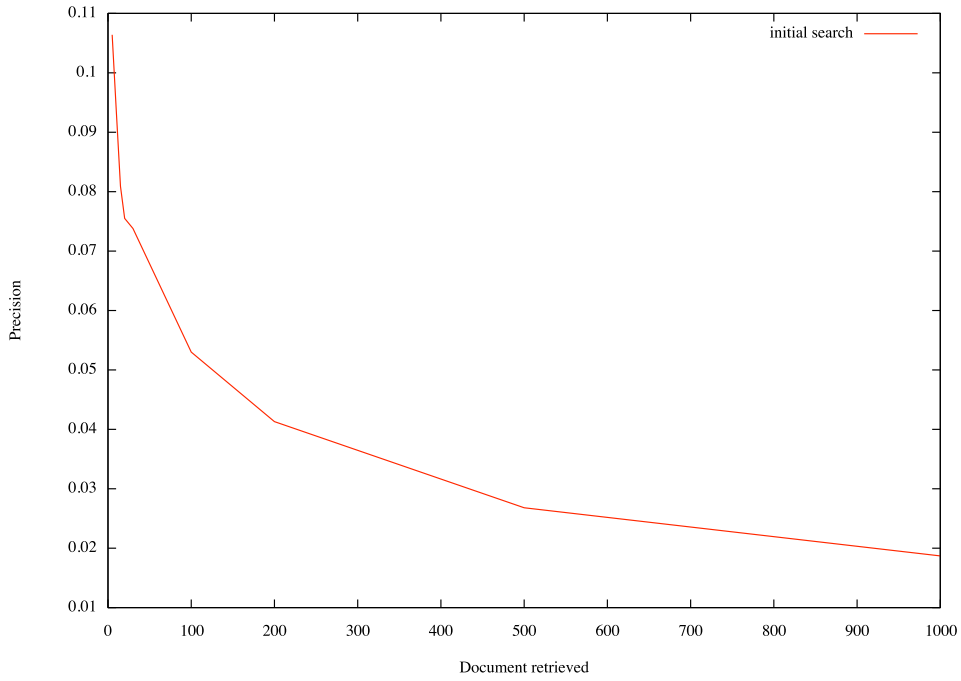

図 4.12: 初期検索の結果

4.5.2 実験・結果

NTCIR の実験では、データセットが非常に大きいこともあり、インデックスの構築においてチューニングは行わず、基本的に TREC と同様に bm25 で行った。ただし、ストップワードを用意してインデックスから削除する代わりに、形態素解析ソフトの出力から自立語だけを用いてインデックスを作成した。

インデックスを作成した結果、初期検索として、図**??**の結果が得られた。TREC の時と同様に、上位 10 件においての精度の分布をとると、図 4.13 という結果が得られた。TREC の時と比べて精度が低い側に分布が偏っている。ただし、本研究の提案手法と既存研究ではどちらも同じインデックスを使うため、比較には問題ないとして続いて比較実験を行った。パラメータは TREC での実験と同じである。

このとき、提案手法と比較手法と初期検索を比べると、図 4.14 の結果が得られた。

上位 100 件までの性能を見たとき、提案手法は比較手法を大きく上回っているが、100 件以下では逆に比較手法のほうが性能が良いことがわかる。

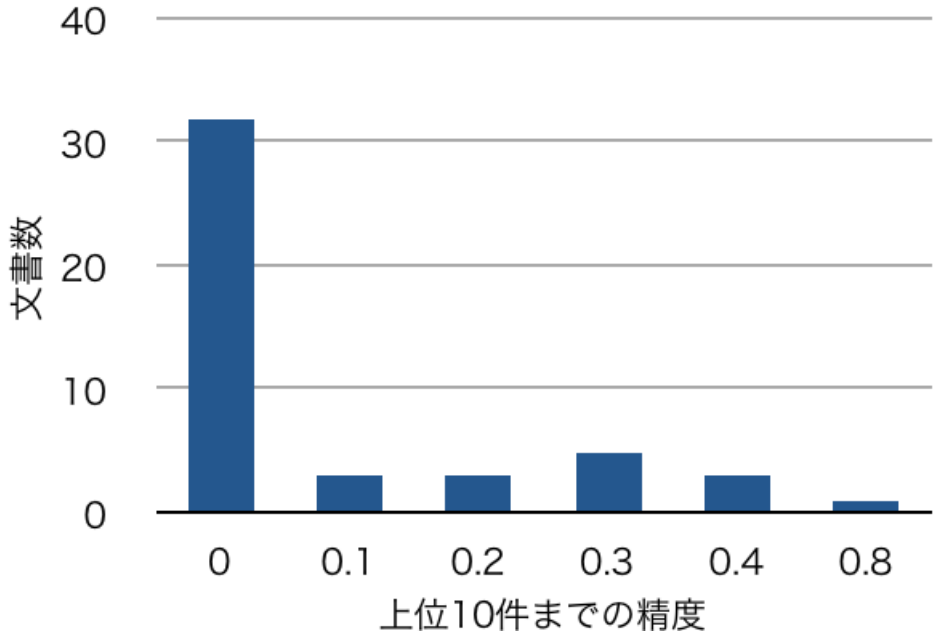

図 4.13: 上位 10 件での精度の分布

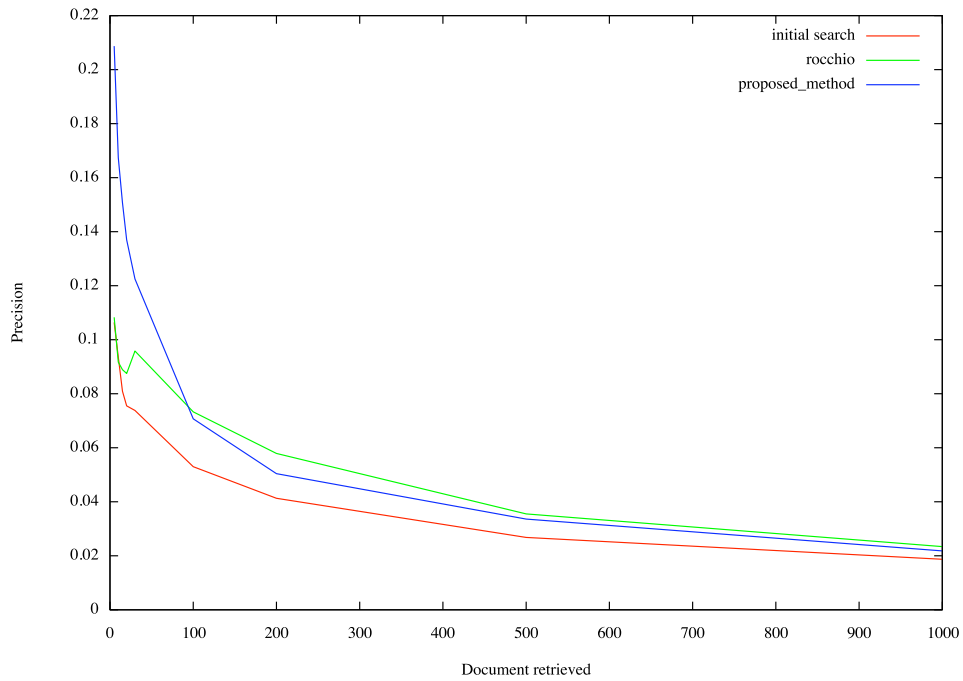

図 4.14: NTCIR を用いた実験結果

4.5.3 考察

本小節では、前小節で得られたパラメータを用いて、NTCIR Web タスクの質問セットを用いて、ベースラインとの比較を行った。

今回の NTCIR を使った実験の初期検索の結果は、ほぼ精度が悪い質問で構成されているため、TREC を用いて実験を行ったときの難しい質問に相当していると考えることが出来る。

図 4.14 からわかるように、上位 100 件までの精度を見ると、提案手法は比較手法を大きく上回っている。一方上位 100 件以下までみると、比較手法を下回ってしまっている。しかし、実際にユーザが検索システムを使う場合、検索結果の 100 件以下までみる場合は少なく、上位 100 件までの精度が向上するというのは非常に有用であると考えられる。

よって、TIPSTER1 のように新聞記事など比較的定型の文書と異なる、WEB のような非定型文書においても、提案手法が有効であることが確認された。また、TREC での実験と同様に、難しい質問について有効であることも確認された。

ただ、インデックスをよりチューニングして初期検索の性能を上げ、上位 10 件の精度が TREC に近いような分布になったとき、初期検索で精度が高い結果が得られるような質問で、本研究の提案手法が比較手法に比べて良い結果が得られるかどうかはわからなかった。

第5章　結論

5.1 研究成果

本研究では、文書検索において、検索結果の改善を行う手法を提案した。

具体的には、検索結果に対してユーザが手動で適合、不適合文書を選ぶ適合性フィードバックにおいて、ユーザがつくりだした適合性判定情報から適当な制約をつくり、その制約を使って検索結果の上位を制約付クラスタリングをすることで、擬似的に適合性判定情報を増加させる手法である。適合性判定情報が増えることにより、フィードバックをして新しい検索結果を得るときに使える情報が増え、より良い結果を得る。

提案手法は TREC の ad hoc 検索タスクと NTCIR の WEB タスクの実験によって有効性の検証を行った。

実験の結果、ユーザが入力した検索クエリに対する検索である初期検索の結果が良い場合、提案手法はベースラインと同程度の結果が得られた。しかし一方初期検索の結果が悪い場合、提案手法は、ベースラインに比べて大きく性能向上した。

検索結果の改善が特に必要となる初期検索の結果が悪い場合に、提案手法は有効であるということが実験によって実証された。

5.2 今後の課題

本研究では実証実験を、一つのクエリに対してある決まったトピックに関する正解データだけが与えられているデータを用いて行った。しかし、前述したように複数のユーザが同じクエリを入力したとしても、同じトピックの情報を求めているとは限らない。

今後の課題としては、同姓同名問題のような複数のトピックが帰ってくるような検索質問に対して、それぞれのトピックごとに正解データを用意し、適合性判定情報を変えることで、トピックごとの検索結果が得られるかどうかを検証することが上げられる。

また、本研究では、検索質問に対して正解データは決まっていることを仮定して人の手を介さずに実験を行ったが、インターフェースなど実際に人が使った場合どう評価するかはわからない。そこで動作するデモシステムを作成し、複数のユーザに利用してもらい結果を評価してもらうことも課題として考えられる。

謝辞

指導教官の安達淳教授には、修士課程の二年間を通じて様々なご指導をいただきました。ご多忙にもかかわらず、丁寧な指導をしていただき、充実した研究生活を送ることが出来ました。本当にありがとうございました。

国立情報学研究所の相沢彰子教授には定期的なミーティングを通じて、研究の指針に関わる様々な助言をいただき、修士研究を進める上で大変お世話になりました。ありがとうございました。

国立情報学研究所の高須淳弘教授には手法のアドバイスをいただくなど、大変お世話になりました。ありがとうございました。

安達研究室の OB の正田 備也さん、OG の若木裕美さん、博士課程 3 年の Vu Quang Minh さん、博士課程 1 年の上村明さん、修士課程 2 年遠山亮介さん、修士課程 1 年の相沢純也さん、修士課程 2 年の倉沢央君、修士課程 1 年の辰巳正治君、修士課程 1 年の広畑堅治君には研究生活において色々とお世話になりました。

みなさん、本当にありがとうございました。

関連図書

[1] インターネット白書. 財団法人インターネット協会, 2006.

[2] *Clusty*, http://clusty.jp/.

[3] IJsbrand Jan Aalbersberg. Incremental relevance feedback. In *SIGIR '92: Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 11–22, New York, NY, USA, 1992. ACM.

[4] Doug Beeferman and Adam L.Berger. Agglomerative clustering of a search engine query log. *KDD*, pp. 407–416, 2000.

[5] I.J. Cox, M.L. Miller, T.P. Minka, T.V. Papathomas, and P.N. Yianilos. The bayesian image retrieval system, pichunter. In *Image Processing, IEEE Transactions on*, 2000.

[6] Douglass Cutting, David Karger, Jan Pedersen, and John W. Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In *Proceedings of the 15th Annual International ACM/SIGIR Conference*, 1992.

[7] Ian Davidson and S. S. Ravi. Agglomerative hierarchical clustering with constraints: Theoretical and empirical results. In *PKDD*, pp. 59–70, 2005.

[8] Harris Drucker, Behzad Shahrary, and David Gibbon. Relevance feedback using support vector machines. In *ICML '01: Proceedings of the Eighteenth International Conference on Machine Learning*, pp. 122–129, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[9] S. Dumais and et al. Sigir 2003 workshop report: Implicit sigir 2003 workshop report: Implicit measures of user interests and preferences. *SIGIR Forum*, 2003.

[10] B. S Everitt. *Cluster Analysis*. Edward Arnold, 1993.

[11] H. Fang, T. Tao, and C. Zhai. A formal study of information retrieval heuristics, 2004.

[12] B. Fuglede and F. Topsøe. Jensen-Shannon Divergence and Hilbert space Embedding. In *Proceedings 2004 International Symposium on Information Theory*, 2004.

[13] Bernard J. Jansen, Amanda Spink, and Tefko Saracevic. Real life, real users, and real needs: a study and analysis of user queries on the web. *Information Processing and Management*, Vol. 36, No. 2, pp. 207–227, 2000.

[14] Glen Jeh and Jennifer Widom. Scaling personalized web search. *WWW*, pp. 271–279, 2003.

[15] Rocchio J.J. *Relevance feedback in information retrieval*, Vol. 313-323. Prentice Hall, Inc, 1971.

[16] D. Klein, S. Kamvar, and C. Manning. From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering, 2002.

[17] Jurgen Koenemann and Nicholas J. Belkin. A case for interaction: A study of interactive information retrieval behavior and effectiveness. In *CHI*, pp. 205–212, 1996.

[18] Jurgen Koenemann and Nicholas J. Belkin. Using relevance feedback and ranking in interactive searching. In *TREC*, 1996.

[19] Robert Krovetz and W. Bruce Croft. Lexical ambiguity and information retrieval. *Information Systems*, Vol. 10, No. 2, pp. 115–141, 1992.

[20] Hisashi Kurasawa, Hiromi Wakaki, Atsuhiro Takasu, and Jun Adachi. Data allocation scheme based on term weight for p2p information retrieval. In *WIDM '07: Proceedings of the 9th annual ACM international workshop on Web information and data management*, pp. 33–40, New York, NY, USA, 2007. ACM.

[21] A. Moschitti. A study on optimal parameter tuning for rocchio text classi er, 2003.

[22] Takashi Onoda, Hiroshi Murata, and Seiji Yamada. One class classification methods based non-relevance feedback document retrieval. In *WI-IATW '06: Proceedings of the 2006 IEEE/WIC/ACM international conference on Web Intelligence and Intelligent Agent Technology*, pp. 393–396, Washington, DC, USA, 2006. IEEE Computer Society.

[23] Guang Qiu, Kangmiao Liu, Jiajun Bu, Chun Chen, and Zhiming Kang. Quantify query ambiguity using odp metadata. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 697–698, New York, NY, USA, 2007. ACM.

[24] Stephen E. Robertson and Karen Sparck Jones. Relevance weighting of search terms. pp. 143–160, 1988.

[25] Stephen E. Robertson, Steve Walker, Micheline Hancock-Beaulieu, Aarron Gull, and Marianna Lau. Okapi at TREC. In *Text REtrieval Conference*, pp. 21–30, 1992.

[26] OZMUTLU Seda, SPINK Amanda, and OZMUTLU Huseyin C. A day in the life of web searching: an exploratory study. *Information processing and management*, 2004.

[27] Craig Silverstein, Hannes Marais, Monika Henzinger, and Michael Moricz. Analysis of a very large web search engine query log. *SIGIR Forum*, Vol. 33, No. 1, pp. 6–12, 1999.

[28] Jaime Teevan, Susan T. Dumais, and Eric Horvitz. Personalizing search via automated analysis of interests and activities. *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 15–19, 2005.

[29] Sergei Vassilvitskii and Eric Brill. Using web-graph distance for relevance feedback in web search. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 147–153, 2006.

[30] Vishwa Vinay, Ken Wood, Natasa Milic-Frayling, and Ingemar J. Cox. Comparing relevance feedback algorithms for web search. *Special interest tracks and posters of the 14th international conference on World Wide Web*, pp. 10–14, May 2005.

[31] Kiri Wagsta, Claire Cardie, Seth Rogers, and Stefan Schroedl. Constrained k-means clustering with background knowledge. In *Proceedings of 18th International Conference on Machine Learning (ICML-01)*, pp. 577–584, 2001.

[32] 邦彦藤田. 徳永健伸 (著), 情報検索と言語処理, 言語と計算シリーズ 5, 東京大学出版会, 1999 年, 3800 円 (税別), isbn4-1-065405-5. 情報処理, Vol. 41, No. 5, p. 606, 20000515.