文書集合からのブール検索式自動生成

岩山 真
(株) 日立製作所中央研究所
makoto.iwayama.nw@hitachi.com

1. はじめに

特許検索などの専門検索では、ブール検索を用いることが多い。検索基準が明確で、絞り込みや拡張の制御が行いやすいためである。システム間で検索基準に違いがないため、構築したブール検索式の汎用性も高い。しかし、望みの文書を検索するには、検索ノウハウや検索対象の領域知識が必要となる。

一方で、文章やその断片を入力し、それに近い文書を検索して提示するランキング検索は、様々な分野で広く普及している。思いついた語句を入力するだけで比較的精度の良い検索が行えるため、特に非専門家にとっては有用な検索法である。しかし、検索基準がわかりにくい、制御が行いにくい、再現性に不安が残る、といった理由から、専門家にはあまり使われていないのが実状である。

本研究の目的は、ブール検索とランキング検索の溝を埋めることである。そのために、任意の文書集合から、それが検索できるブール検索式を逆生成する手法を提案する。ランキング検索の結果が等価なブール検索式に変換できれば、まずはランキング検索を行い、有望な結果が得られたら、そこからブール検索式を自動生成させ、必要であれば修正してブール検索に移行することができる。また、生成させたブール検索式をランキング検索の根拠とみなすこともできる。

自動生成させたブール検索式を介して、書誌情報の差異を吸収することもできる。例えば、特許検索では、国際特許分類(IPC)による分類検索を行うことが多い。しかし、論文など特許以外の文書には国際特許分類は付与していない。このような場合でも、国際特許分類で検索した特許集合からキーワードのブール検索式を生成すれば、その検索式を論文検索システムに入力することで、分類検索と等価な検索が行える。ほとんどの検索システムはキーワードで構成されたブール検索式を受け付けるため、ブール検索式を媒体として任意のシステム間で検索結果を引き継ぐことが可能になる。

2. 決定木に基づく方法（従来法）

Kim らは、与えられた文書集合から、決定木によりブール検索式を生成する手法を提案した[1] 。与えられた文書集合が正例となり、それ以外の文書集合が負例となる。正例、負例から、それらを弁別する決定木を C4.5[3] で学習する。図 1 に学習した決定木の例を示す。

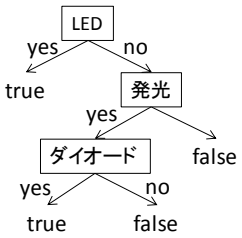

図 1: 決定木による正例と負例の弁別

決定木から、真(true)の値を持つ葉ノードを集め、それぞれに対し根ノードからの経路を論理積で結合する。これらの論理積の論理和が求めるブール検索式になる。図 1 の決定木からは、「LED∨(発光∧ダイオード)」が生成される。Kim らの研究では、検索式に含まれるタームの候補集合を変えて多数の論理積を生成し、ランク学習で最適な論理積集合を選択しているが、本稿では、前段の決定木の部分のみを考察する。

決定木に基づく方法の問題点は、負例の与え方が難しいことである。与えられた文書集合以外の文書は全て負例になるため、負例の数は膨大である。よって、サンプリングが必要になるが、最適なサンプリング方法、サンプリング数を決めるのは容易でない。

特に、適切な負例を与えないと、共起するタームを論理積として生成できない。例えば、「発光」「ダイオード」双方を含む文書の集合が正例になる場合、負例としてどちらか一方のみを含む文書も与えないと、「発光∧ダイオード」という論理積が生成できない。「発光」もしくは「ダイオード」のみでも正例と負例が弁別で

きてしまうためである。正解の論理積に対して、常にこのような負例を与えることは困難である。

3. 被覆アルゴリズムに基づく方法

前節で考察したように、検索すべきでない文書は膨大にあるため、それら負例を必要十分にサンプリングするのは難しい。そこで本研究では、正例と負例の弁別に基づく決定木のような方法ではなく、正例の被覆(covering)に基づく繰り返しアルゴリズム[2] を採用する。被覆の目的関数には F 値を用いる。負例を与えなくて良い反面、大域的な情報として各タームのヒット件数が必要になる。

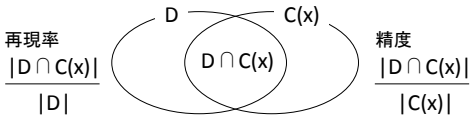

図 2: 与えられた文書集合とブール検索式との関係

本研究では、与えられた文書集合が、F 値における正解となる。その正解を漏れなく(再現率が高く)、かつ、ノイズなく(精度が高く)検索できるのが理想のブール検索式である。与えられた文書集合をD、求めるブール検索式をx、検索対象内でxにヒットする文書集合を$C(x)$とすると、これらの関係は図 2 で表すことができ、再現率と精度の調和平均である F 値 $f(x)$は以下の式で計算できる。

$$f(x) = \frac{2|D \cap C(x)|}{|D| + |C(x)|} \qquad (1)$$

ここでの目的は、$f(x)$を最大化するxを探索することである。提案手法では、xを積和標準形$x_1 \vee x_2 \vee \cdots$で表し、それぞれの構成論理積x_iを貪欲法で順番に探索していく。よって、全体として F 値が最大となる積和標準形が探索できるわけではない。ある論理積x_iを探索した後は、対象の文書集合(初期値はD)からx_iにヒットする文書を除いて次の論理積x_{i+1}を探索する。文書集合が空になるか、新たにヒットする文書数がある閾値を下回ったら探索を終了する。後者は、過適合を防ぐための条件である。以上の処理手順をアルゴリズム 1 に示す。

各構成論理積$x_i = y_{i,1} \wedge y_{i,2} \wedge \cdots$は、山登り法で探索する。よって、ここでも最適性は保証されていない。まず、F 値が最大となるターム$y_{i,1}$を出発点とする。次に、論理積で結合した後の F 値が最大となるターム$y_{i,2}$を探索する。このようにして、あらかじめ決められた長さまで論理積を伸ばしながら F 値が最大の論理積を探索する。

アルゴリズム 1: 積和標準形探索アルゴリズム

```
入力：文書集合D, 論理積の最小ヒット数 th
DNF =null
while true
    if  |D| < th  then break
    x = argmax_x f(x)  s.t. xはタームの論理積
    if |C(x)| < th then break
    DNF = DNF ∨ x
    D = D\C(x)
end while
出力：DNF
```

式(1)で論理積xの F 値を計算する際、$|D \cap C(x)|$は与えられた文書集合D内での論理積xのヒット件数であるから、その場で集計できる。一方、$|C(x)|$は検索対象内での論理積xのヒット件数であるから、実際に検索してみないとわからない。本研究では、タームが独立に出現することを仮定し(文書分類のナイーブベイズ法と同様の仮定)、論理積xのヒット件数を、構成タームのヒット件数から以下の式で推定する。

$$|C(x = y_1 \wedge y_2 \wedge \cdots)| \approx \max\left\{|D \cap C(x)|,\ \ |C| * \prod_i \frac{|C(y_i)|}{|C|}\right\} \qquad (2)$$

ここで、$|C|$は検索対象の総文書数である。また、検索対象Cでのヒット件数を、与えられた文書集合Dでのヒット件数$|D \cap C(x)|$で平滑化している。平滑化後は論理積を伸ばしても F 値が単調減少するため、平滑化した時点で探索を打ち切れる。

論理積の探索手順をアルゴリズム 2 に示す。山登り法で探索を行うため、複数の出発点から探索を行うことで局所最適に陥ることを防いでいる。

アルゴリズム 2: 論理積探索アルゴリズム

```
入力：文書集合D, 論理積の最大長 len, 初期点数 b
Conj =null
for i = 1 to b
    x = y  s.t. yはf(y)が i 番目に大きいターム
    for j = 1 to len
        if f(x) > f(Conj) then Conj = x
        if f(x)の計算時に平滑化した then break
        if j < len then
            y = argmax_y f(x ∧ y)
            x = x ∧ y
        end if
    end for
end for
出力：Conj
```

提案手法を実際に使う際の注意点について述べる。

- タームのヒット件数が取得できない検索エンジンも多い。その場合は、別のコーパスで取得したヒット件数(出現確率)を転用する。ただし、目的の検索エンジンが持つ総文書数|C|は必須である。
- 検索させたい文書を全てDとして数え上げられないことも多い。そこで、仮想の目標母集団PからDがランダムサンプリングされたと仮定して、F 値を以下のように一般化する。

$$f(x) = \frac{2|D \cap C(x)|}{|D| + (|D|/|P|)|C(x)|} \qquad (3)$$

母集団の大きさ|P|を目標ヒット件数のパラメータと解釈すれば、|P|を変えることで様々な粒度のブール検索式を生成することもできる。

4. 実験

4.1. ブール検索式の復元

まず、人工的な実験により提案手法を評価した。あるブール検索式から検索を行い、検索結果からブール検索式を生成し、検索に用いたブール検索式と比較する。ブール検索式の復元を試みていることになる。

実験では、特許検索のログから抽出した 100 個のブール検索式を用いた。ただし、タームは形態素に限った。半数以上の 57 個は 2 つのタームからなる単純な検索式であるが、「(放熱∨(熱∧伝導)∨(伝∧熱))∧シート」といった複雑なものもある。検索式あたりの延べターム数の平均は 3.23 個である。

検索対象は、1993 年から 2007 年に公開された特許公開公報の要約部分であり、検索結果から公開日が古い順に 100 件抽出して検索式を生成させた。検索件数が 100 件に満たない場合は全件用いている。検索件数の平均は 17,574 件なので、ほとんどのケースで式(3)により F 値を計算することになる。

比較する手法は、決定木(C4.5)に基づく方法と提案手法である。決定木では、正例と同数の負例をランダムに選んで与えた。提案手法では、論理積の最大長lenを 5、初期点の数bを 10、論理積の最小ヒット数thを$|D| \times 0.033$とした。いずれの手法も正例から抽出した全ての形態素を候補タームとした。

正解のブール検索式と生成したブール検索式との一致度は、以下の方法で計算した。まず、正解のブール検索式を積和標準形に変換する。生成したブール検索式は、決定木、提案手法共に積和標準形である。次に、正解の積和標準形の各論理積Aに対し、以下の値が最大となる、比較対象中の論理積Bを見つける。ただし、重複は許さない。

$$\frac{\text{A と B が共通に含むターム数}}{\max\{\text{A のターム数},\ \text{B のターム数}\}}$$

全ての A に渡ってこの値を足し、A の総数、Bの総数の大きい方で割った値を一致度とする。両者のブール検索式が完全に一致すれば、一致度は最大の 1 になる。

表 1: 一致度(平均値)の比較

	提案手法	決定木
一致度	0.8687	0.5276

表 1 に実験結果を示す。提案手法では、2 個のタームから成る単純な検索式については、全てが完全に復元できた。複雑な検索式もほぼ問題なく復元できている。例えば、前述の「(放熱∨(熱∧伝導)∨(伝∧熱))∧シート」に対しては、「(放熱∧シート)∨(伝導∧シート)∨(伝∧熱∧シート)」を生成した(一致度 0.89)。ここで、完全に復元できていない論理積は、「熱∧伝導∧シート」に対する「伝導∧シート」であるが、この原因は正例の不足にある。総ヒット件数 2210 件中で、「熱∧伝導∧シート」が 1356 件、「伝導∧シート」が 1371 件にヒットしていたのに対し、選んだ 100 件中では、両者 69 件とヒット件数に差がなくなってしまった。また、「スパム∨SPAM∨ジャンク∨迷惑∨悪戯∨いたずら∨嫌がらせ」のように、多数のタームの論理和から成るブール検索式を復元する場合、100 件の正例にこれらのタームが全て現れずに復元しきれない事例も多かった。例の場合は、「ジャンク∨迷惑∨悪戯∨いたずら」までしか復元できなかった(一致度 0.57)。

一方、決定木は、2 節で考察したように論理積の復元が難しい。2 個のタームから成る単純な論理積 52 個のうち完全に復元できたのは 8 個のみであった。これは負例を増やせば改善できるため、与える負例を正例の N 倍に増やしてみた。実験結果を表 2 に示す。10 倍までは負例の数が増すにつれ一致度も上がるが、それ以降は、多数の負例の影響で、逆に一致度が下がった。このことからも、決定木では負例の与え方が難しいことがわかる。

表 2: 決定木における負例数の影響

負例数	x1	x5	x10	x30	x50
一致度	0.5276	0.6211	0.6615	0.6403	0.6068

提案手法では、F 値が最大の論理積を山登り法で探索する。よって、探索の出発点が少ないと局所最適に陥りやすい。表 3 に、出発点の数と一致度との関係を示す。表より、出発点の数は 5 で十分なことがわかる。

表 3: 論理積探索における出発点の数の影響

出発点数	1	3	5	10	20
一致度	0.8100	0.8473	0.8689	0.8687	0.8692

また、提案手法では、大域情報としてタームのヒット件数が必要である。目的のコーパスでのヒット件数が取得できない場合は、別のコーパスでのヒット件数(出現確率)で近似することもできる。そこで、ヒット件数の情報源として論文コーパス(NTCIR-1,2)を用いて実験してみると、一致度は 0.7330 に下がった。

4.2. ランキング検索結果からの生成

本実験では、ランキング検索の結果からブール検索式を生成させて評価した。ランキング検索の入力には、NTCIR-4 の特許検索タスクのトピック 34 件を用いた。具体的には特許請求項が検索の入力文章となる。

検索モデルにはベクトル空間モデルを用いた。検索対象は 1993 年から 1997 年に公開された特許公開公報の要約部である。検索結果の上位 100 件を正例としてブール検索式を生成させた。決定木では、101 位から 200 位までの 100 件を負例として与えた。これは Kim らの手法と同じ与え方である。提案手法のパラメータは前節の実験と同じである。

表 4: 入力文書集合の再現率と精度

	提案手法	決定木	決定木(ランダム負例)
再現率	0.9271	0.7718	0.9944
精度	0.0571	0.0627	0.0063

まず、生成したブール検索式で正例 100 件がどの程度漏れなくかつノイズ無く検索できるかを調べる。表 4 に、正例を正解とした時の再現率と精度を示す。表からわかるように、両者、再現率は十分高いが精度が極端に低い。つまり、与えた 100 件のみを検索するブール検索式を生成することは難しい。そもそもそのようなブール検索式は存在しないかもしれない。決定木は、精度で提案手法を若干上回るが、これは 101 位から 200 位までのニアミスを負例として与えたためである。ランダムに負例を与えると、精度が一桁悪化した。

生成したブール検索式の例を一つ示す。

> 箱状の成形基台に開放された凹所を形成し、前記凹所の内面を含む成形基台の表面に所定のパターンの導電膜を形成し、前記凹所にセンサ用素子を接合し、成形基台を蓋で密閉してなることを特徴とするセンサ装置。

この請求項の検索結果から、提案手法は以下のブール検索式を生成した。

(凹∧所∧成形∧有する) ∨ (凹∧形成∧導∧電∧素子) ∨ (凹∧部∧基∧台∧設ける) ∨ (凹∧所∧形成) ∨ (凹∧所∧台∧設ける)

一方、決定木は以下を生成した。

(基∧形成) ∨12 ∨ 21

3 節で述べたように、提案手法では、目標ヒット件数をパラメータ(あくまでも目標値)として与えることができる。例えば、上記の例で、目標ヒット件数を変えてブール検索式を生成した結果を以下に示す。

100,000	10,000	1,000
凹∨ (基∧台)	(凹∧導∧電)∨ (凹∧所)∨ (基∧台∧部)∨ (形成∧成形∧12)	(凹∧導∧電∧素子)∨ (凹∧所)∨ (基∧台∧凹∧部)∨ (台∧載る∧置)

目標ヒット件数が減るにつれ、論理積に新たなタームが付加されていく。このような情報は検索結果の絞り込み支援でも役に立つ[4] 。

5. おわりに

与えられた文書集合を出来るだけ正確に検索できるブール検索式を自動生成する方法を提案して評価した。否定の導入、タームとして任意の部分文字列を許すこと、などが今後の課題である。

参考文献

[1] Y. Kim, J. Seo, and W.B. Croft, “Automatic Boolean Query Suggestion for Professional Search”, In Proc. of SIGIR ’11, 2011.

[2] M. Kubat, I. Bratko, and R. Michalski, “A Review of Machine Learning Methods”, in Machine Learning and Data Mining, pp.3-69, John Wiley &Sons, 1996.

[3] J.R. Quinlan, “C4.5: Programs for Machine Learning”, Morgan Kaufmann, 1993. (http://www.rulequest.com/Personal/c4.5r8.tar.gz)

[4] 松生, 是津, 小山, 田中, “検索結果の概要を表すキーワード式生成による質問修正支援”, DEWS2005, 1C-i9, 2005