ランダム・プロジェクションを用いた情報検索システム

佐々木 稔　北 研二
徳島大学工学部

1 はじめに

近年，インターネットの普及とともに，個人でWWW (World Wide Web) を代表とするネットワーク上の大量の電子データやデータベースが取り扱えるようになり，テキストデータの山の中で必要な情報を取り出す機会が増加している．しかし，このようなデータの増加は必要な情報の抽出を困難とする原因となる．この状況を反映し，情報検索，情報フィルタリングやクラスタリングといった技術が注目を集め，過去数十年の間に新聞記事などの文書を対象とした研究が盛んに進められ，高速な文字列検索アルゴリズムや自動索引づけなどに多くの成果が得られている．

このような情報検索システムのなかでよく使われる検索モデルとして，ベクトル空間モデル [1] がある．これは，文書と検索要求がベクトルを用いて，多次元空間内の点として表現されるもので，基本的には文書集合から索引とするタームを取り出し，タームはベクトルの要素として数値的に保存される．このとき，タームに重みが加えられることにより，文書全体に対するタームの特徴を目立たせることが可能で，この重みを計算するために，IDF (Inverse Document Freqency)[2] などのような重みづけ方法が数多く提案されている．また，ベクトルを比較する類似度の尺度として，内積やなす角の余弦尺度がよく用いられている．類似度計算により,, 類似度の高いものからランクづけを行い，ユーザに表示することができることもこのモデルの特徴のひとつである．

新聞記事などの大量の文書データをベクトルにする場合，その要素となるタームの数が非常に多くなり，ベクトルは高い次元を持つようになる．しかし，ひとつの文書データに存在するタームの数は全体のターム数に比べると非常に少なく，文書ベクトルの要素には 0 が多い，すなわちスパースなベクトルになる．このようなベクトルを用いて類似度を計算する際には，計算量が多いことによる検索時間の長さや計算に必要なメモリの量が大きな問題となる．このため，単語の意味，相互関係などの情報を用いたり，ベクトル空間モデルの構造を利用してベクトルの次元を圧縮することでこれらの問題が解消されると考えられる．これまでにも，ベクトル圧縮技術として，統計的なパターン認識技術や線形代数を用いたさまざまな次元圧縮技術が研究され，特異値分解 [3], semi-discrete matrix decomposition[4] を用いた Latent Semantic Indexing (LSI)[5], 主成分分析を応用した Karhunen-Loève 変換を用いて次元圧縮を行う Fast Map[6] などがある．これらの手法を用いることにより，次元圧縮を行わないモデルに比べて良い性能が示されているが，検索モデルを構築する時間や検索時間が実用に耐えられないことが問題となっている．

本稿では，ベクトル空間モデルの次元圧縮手法として，検索時間削減が主な目的であるランダム・プロジェクション [7] を用いた検索モデルを提案する．ランダム・プロジェクションは，行列を任意の低い次元の部分空間に射影するとき，LSI を用いて次元圧縮を行った場合と同様に距離を保存しやすい特性を持っている．さらに，LSI を用いて行列の次元圧縮をした場合に比べ，モデルの性能を下げることなく計算量を大幅に削減することができると考えられる．ランダム・プロジェクションを用いた検索モデルを構築し，評価用テストコレクションである MEDLINE を利用した検索実験を行い，本提案手法の有効性を示す．

2 ランダム・プロジェクション

本節においては，ランダム・プロジェクションを用いたベクトル空間モデルについて述べる．ひとつの文書データは n 次元空間 $\mathbf{R}^n$ 上の点として表され，概念ベクトルはいくつかの文書データからなる n 次元空間内にある点の集合とする．この n 次元空間内のベクトル $u \in \mathbf{R}^n$ が k 次元空間に射影される際，まず k 個の任意の n 次元ベクトル $r_1, \cdots, r_k$ を用意する．これらのベクトルともとの n 次元ベクトルの

内積 $u'_1 = r_1 \cdot u, \cdots, u'_k = r_k \cdot u$ を計算し，それらの値を要素とする k 次元ベクトルを計算する．ここで，行列 R をその列ベクトルが $r_1, \cdots, r_k$ に等しい $n \times k$ の行列とすると，求める k 次元ベクトルは $u' = R^T u$ となる．$\mathbf{R}^n$ の点 $u_1, \cdots, u_m$ を $\mathbf{R}^k$ 空間に射影するためには，行列 R が得られることで，ランダム・プロジェクションは行列の計算 $R^T u_1, \cdots, R^T u_m$ を行うだけの簡単な計算で表すことができる．この行列 R が任意の正規直交行列のとき，すなわち行列 R の列ベクトルが単位ベクトルで，そのふたつの列ベクトルが直交すれば，この射影は距離を保存する特性を持っている．

この行列 R を求めるために，クラスタ構造を求めることができる球面 k 平均アルゴリズム [9] を用いる．これは，文書データの集合などで作られる高い次元でスパースな文書ベクトルをクラスタリングするためのアルゴリズムで，構造化されていない文書集合にある隠れた概念を見つけるために用いられる．この球面 k 平均アルゴリズムについては，3 節で述べる．

次に，ランダム・プロジェクションが LSI にくらべて計算量がどれだけ短縮されているのかを示す．A は $n \times m$ の行列とする．このとき，A の列ベクトルに平均して c 個の 0 でない要素が存在すると，LSI を用いた場合の計算量は $O(mnc)$ となる．また，l 次元に縮小するランダム・プロジェクションでは，$O(mcl)$ の計算量が必要となる．ランダム・プロジェクションの後，索引付けを行ったときの計算量は $O(ml^2)$ で，全体で圧縮するのに必要な計算量は $O(ml(l+c))$ となる．先の不等式に示した ϵ 近似を得るために，$O((\log n)/\epsilon^2)$ となる l を用いると，全体の計算量は $O(m(\log^2 n + c \log n))$ で，通常の LSI の $O(mnc)$ に比べて漸近的に優れていることがわかる．

3 球面 k 平均アルゴリズム

ここでは，ランダム・プロジェクションに必要な行列を求めるために，高い次元でスパースな文書データ集合をクラスタリングするために用いられる球面 k 平均アルゴリズム [9] について述べる．このアルゴリズムは，ユークリッド空間内でのなす角の余弦を類似度とし，多次元空間の単位円を分割することによりクラスタリングを行うものである．これにより，文書ベクトルの集合は指定した数の部分集合に分割され，各集合の中心を計算することで，概念ベクトルを作ることができる．さらに，このアルゴリズムは文書ベクトルのスパースさを逆に利用して，局所解に早く収束するという利点がある．まず，球面 k 平均アルゴリズムの概要を述べる前に，クラスタリングをして得られる概念ベクトルとこのアルゴリズムの目的関数を定義する．

n 個のベクトル $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n$ が存在し，これらのベクトルはすべて正規化されている．また，$\pi_1, \pi_2, \cdots, \pi_k$ は，ベクトルを異なる k 個の集合にクラスタリングしたベクトルの集合とする．このとき，各 π_j に含まれるベクトルの平均である，重心は，

$$\mathbf{m}_j = \frac{1}{n_j} \sum_{\mathbf{x} \in \pi_j} \mathbf{x}$$

となる．ここで n_j は π_j に含まれるベクトルの数を表す．ベクトルの重心は単位長にはなっていないので，そのベクトルの長さで割ることにより概念ベクトル $\mathbf{c}_j$ が定義される．

$$\mathbf{c}_j = \frac{\mathbf{m}_j}{\|\mathbf{m}_j\|}$$

概念ベクトルは任意の単位ベクトル $\mathbf{z}$ に対しても，以下のようにコーシー・シュワルツの不等式が成り立つ．

$$\sum_{\mathbf{x} \in \pi_j} \mathbf{x}^T \mathbf{z} \leq \sum_{\mathbf{x} \in \pi_j} \mathbf{x}^T \mathbf{c}_j$$

概念ベクトルは，π_j に含まれるすべてのベクトルとの余弦が最も近いベクトルであると考えることができる．

この不等式から，各クラスタ $\pi_i (1 \leq j \leq k)$ の密度を $\sum_{\mathbf{x} \in \pi_j} \mathbf{x}^T \mathbf{c}_j$ と定義する．先に示したように，$\sum_{\mathbf{x} \in \pi_j} \mathbf{x} = n_j \mathbf{m}_j$，$\|\mathbf{c}_j\| = 1$ が成り立つので，以下に示すようにクラスタ π_j の密度はこのクラスタに属するベクトルの和の距離に等しくなる．

$$\begin{aligned} \sum_{\mathbf{x} \in \pi_j} \mathbf{x}^T \mathbf{c}_j &= n_j \mathbf{m}_j^T \mathbf{c}_j = n_j \|\mathbf{m}_j\| \mathbf{c}_j^T \mathbf{c}_j \\ &= n_j \|\mathbf{m}_j\| = \| \sum_{\mathbf{x} \in \pi_j} \mathbf{x} \| \end{aligned}$$

これより，球面 k 平均アルゴリズムを用いて得られるベクトルの分割に対する密度は k 個のクラスタの結合密度の和として，次の目的関数 $Q(\{\pi_j\}_{j=1}^k)$ が定義される．

$$Q(\{\pi_j\}_{j=1}^k) = \sum_{j=1}^{k} \sum_{\mathbf{x} \in \pi_j} \mathbf{x}^T \mathbf{c}_j$$

3.1 球面 k 平均アルゴリズム

目的関数 $Q(\{\pi_j\}_{j=1}^k)$ を最大にするように，ベクトルの集合をクラスタリングする球面 k 平均アルゴリズムについて述べる．その例として，文書ベクトル $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n$ を k 個のクラスタ $\pi_1^\star, \pi_2^\star, \cdots, \pi_k^\star$ に分割する場合を考える．

この目的関数の最適解を求めるためのアルゴリズムを以下に示す．

1. すべての文書ベクトルを k 個のクラスタに任意に分割する．これらの部分集合を $\{\pi_j^{(0)}\}_{j=1}^k$ とし，これより求められた概念ベクトルを $\{\mathbf{c}_j^{(0)}\}_{j=1}^k$ とする．

2. 各文書ベクトル $\mathbf{x}_i (1 \leq i \leq n)$ に対し，$\mathbf{x}_i$ とのなす角の余弦が最も近い概念ベクトルを見つける．これにより，前の概念ベクトル $\{\mathbf{c}_j^{(t)}\}_{j=1}^k$ から文書ベクトルが新たな部分集合 $\{\pi_j^{(t+1)}\}_{j=1}^k$ に分割される．

$$\pi_j^{(t+1)} = \{\mathbf{x} \in \{\mathbf{x}_j\}_{i=1}^n : \mathbf{x}^T \mathbf{c}_j^{(t)} \geq \mathbf{x}^T \mathbf{c}_l^{(t)}, \quad 1 \leq l \leq n\} \ (1 \leq j \leq k)$$

ここで，$\pi_j^{(t+1)}$ は概念ベクトル $\mathbf{c}_j^{(t)}$ に近いすべての文書ベクトルの集合とする．

3. 新たに導かれた概念ベクトルの長さを正規化する．

$$\mathbf{c}_j^{(t+1)} = \frac{\mathbf{m}_j^{(t+1)}}{\|\mathbf{m}_j^{(t+1)}\|} (1 \leq j \leq k)$$

ここで，$\mathbf{m}_j^{(t+1)}$ はクラスタ $\pi_j^{(t+1)}$ の文書ベクトルの重心を表す．

4. 停止基準を超えると，$\pi_j^\star = \pi_j^{(t+1)}$，$\mathbf{c}_j^\star = \mathbf{c}_j^{(t+1)}$ $(1 \leq j \leq k)$ となり，このアルゴリズムは終了する．停止基準を超えていない場合は，t に 1 が加えられて，ステップ 2 に戻る．

停止基準の例としては，

$$|Q(\{\pi_j^{(t)}\}_{j=1}^k) - Q(\{\pi_j^{(t+1)}\}_{j=1}^k)| \leq \epsilon$$

を適当な $\epsilon > 0$ に対して調べるなどがある．

4 実験

ランダム・プロジェクションを用いた検索モデル構築し，その検索性能を示す実験を行った．実験には，情報検索システムの評価用テストコレクションである MEDLINE を利用した．MEDLINE は医学・生物学分野における英文の文献情報データベースで，検索の対象となる文書の件数は 1033 件で，約 1Mbyte の容量を持つテキストデータである．また，MEDLINE には 30 個の評価用検索要求文とそれらの関連記事が用意されている．

まず，前処理として MEDLINE の記事全体から抽出した 1033 件の記事から一般的な 439 個の英単語をストップワードに指定して，文書の内容と関係のほとんどない単語は削除した．この前処理の結果，4329 個のタームが索引語として抽出された．

これらの索引語を要素とする文書ベクトルを作成するとき，索引語の頻度に重みを加えた数値をベクトルの要素とする．数多く提案されている重みづけ手法で，今回の実験では以下の式で定義された対数エントロピー重み [2] を用いた．L_{ij} は j 番目の文書に対する i 番目のタームへの重み，G_i は文書全体に対する i 番目のタームへの重みを表す．

$$L_{ij} = \begin{cases} 1 + \log f_{ij} & (f_{ij} > 0) \\ 0 & (f_{ij} = 0) \end{cases}$$

$$G_i = 1 + \sum_{j=1}^{n} \frac{\frac{f_{ij}}{F_i} \log \frac{f_{ij}}{F_i}}{\log n}$$

ここで，n は全文書数，f_{ij} は j 番目の文書に出現する i 番目のタームの頻度，F_i は文書集合全体における i 番目のタームの頻度を表す．

得られた文書ベクトルから，球面 k 平均アルゴリズムを用い，これらの文書ベクトルより指定された数の概念ベクトル作成する．作成した概念ベクトルを結合した行列に対し，ランダム・プロジェクションを行い，文書ベクトル，検索要求ベクトルの次元を削減する．次元の削減されたベクトルに対し，内積の計算を行い，その値を各文書ベクトルへの検索スコアとする．これらのスコアのうち，上位 50 文書を検索結果として，出力する．

検索システムの精度の評価には，一般的に用いられている正解率 (Precision) と再現率 (Recall) を用いた [10]．再現率と正解率は，それぞれ個別に用いて，システム評価を行うことができるが，本実験では，一般にランクづけ検索システムの評価に用いられる再現率-正解率曲線を用い，システムの評価を行った．この曲線は，各質問に対しひとつの曲線が作成されるが，本稿の検索システム評価には，全質問に対し各再現率での平均を計算した再現率-正解率曲線と，検索した文書の数に従って，平均正解率がどの

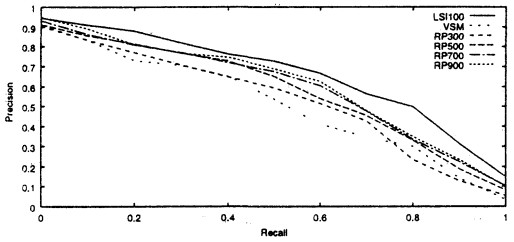

図 1: 再現率-正解率曲線

ように変化しているのか示す文書数-正解率曲線を用いた.

本実験では，検索モデルとして，ランダム・プロジェクションを行い，ベクトルの次元をそれぞれ 300, 500, 700, 900 に圧縮したものと，これらと比較するために，次元圧縮していない元のベクトル空間モデルと LSI を用いて次元数を 100 に圧縮したものについても検索実験を行った．この結果，検索要求 30 個の検索結果の平均を求めた再現率-正解率曲線は図 1 になり，LSI と同じ程度に検索精度が改善されたことが示されている．

また，検索モデルを作成するために必要な時間は，ランダム・プロジェクションを用いた場合は約 7 分で，LSI の場合は約 24 分となり，ランダム・プロジェクションは LSI に比べ高速に検索モデルを構築することができる．ひとつの検索要求に対する検索時間は，ランダム・プロジェクションを用いた場合は約 4 分必要で，LSI や元のベクトル空間モデルの約 3 秒に比べて大幅に検索時間が必要であった．これは，検索時にランダム・プロジェクションにより文書ベクトルの次元圧縮が行われているためで，LSI と同様に検索前にあらかじめ次元圧縮を行うことにより，検索時間が短縮できると考えられる．

5 おわりに

本稿では，ベクトル空間モデルの次元圧縮手法として，ランダム・プロジェクションを用いた検索モデルを提案し，検索実験を行った．その結果，次元圧縮していない元のベクトル空間モデルと比べ検索精度が改善され，ランダム・プロジェクションが有効な検索手法であることが示された．また，LSI に対しても，ランダム・プロジェクションを用いた検索モデルとの検索精度の差は少なく，ランダム・プロジェクションが LSI と同じくらいの次元圧縮性能を持っていることが示された．

今後の課題としては，本実験で用いたテスト・コレクション MEDLINE には収録されているデータの少なさが問題となる．このため，日本語情報検索の評価用テスト・コレクションである BMIR-J2[11] のような大規模なテスト・コレクションを用いた検索実験，および評価をすることが必要であると考えている．また，ランダム・プロジェクションを用いて次元圧縮をする行列を求めるために，球面 k 平均アルゴリズムを用いたが，その有効性を示すことも必要である．様々なクラスタリング手法や単純な正規分布により得られた行列などを用いて比較を行い，球面 k 平均アルゴリズムが有効であるかを検証したい．

参考文献

[1] Gerard Salton, Michael J. McGill: "Introduction to Modern Information Retrieval", McGraw-Hill, 1983.

[2] Erica Chicholm, Tamara G. Kolda: "New Term Weighting Formulas for the Vector Space Method in Information Retrieval", Technical Memorandum ORNL-13756, Oak Ridge National Laboratory, Oak Ridge, Tennessee, 1998.

[3] Scott Deerwester, Susan T. Dumais, Thomas K. Landauer, George W Furnas, Richard A. Harshman: "Indexing by Latent Semantic Analysis", Journal of the Society for Information Science, 41(6), pp. 391-407, 1990.

[4] Tamara G. Kolda, Dianne P O'Leary: "A Semidiscrete Matrix Decomposition for Latent Semantic Indexing in Information Retrieval", Proceedings of ACM Transaction on Information Systems, Vol. 16 pp. 322-346, 1998.

[5] Susan T. Dumais: "Using LSI for information filtering: TREC-3 experiments", D. Harman (Ed.), Overiew of The Third Text REtrieval Conference (TREC3) National Institute of Standards and Technology Special Publication 500-226, pp. 219-230, 1995.

[6] Christos Faloutsos, King-Ip Lin, "FastMap: A Fast Algorithm for Indexing, Data-Mining and Visualization of Traditional and Multimedia Datasets", Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data, pp. 163-174, May 1995.

[7] Christos Papadimitriou, Prabhakar Raghavan, Hisao Tamaki, Santosh Vempala: "Latent Semantic Indexing: A Probabilistic Analysis", Proceedings of the 17th ACM Symposium on the Principles of Database Systems, pp. 159-168, 1998.

[8] Rosa I. Arriaga, Santosh Vempala: "An algorithmic theory of learning: Robust Concepts and Random Projection", Proceedings of the 40th Foundations of Computer Science, 1999.

[9] Inderjit S. Dhillon, Dharmendra S. Modha: "Concept Decompositions for Large Sparse Text Data using Clustering", Technical Report RJ 10147, IBM Almaden Research Center, 1999.

[10] David D. Lewis: "Evaluating Text Categorization", DARPA, February 1991.

[11] 木谷 強ほか: "日本語情報検索システム評価用テストコレクション BMIR-J2", 情報処理学会研究会報告 98-DBS-114-3, pp. 15-22, 1998.