[論文番号：C2-2]

潜在的意味抽出方式と意味の数学モデルによる意味的連想検索方式の比較

伊東拓†，中西崇文††，北川高嗣†††，清木康††††

概要

現在，膨大な情報がコンピュータネットワーク上に散在している．このような状況においては，的確に必要な情報を得る方式が重要であり，現在までに幾つかの方式が提案されてきた．これらの方式の中で，本稿では潜在的意味抽出方式(Latent Semantic Indexing method, LSI)と，我々が提案している意味の数学モデルによる意味的連想検索方式の２つについて比較検討を行う．両方式は，ある空間内でデータをベクトル表現し，比較を行うという方式である．しかしながら，LSIでは静的な空間を用い，意味の数学モデルによる意味的連想検索方式では，文脈に応じて動的な空間を構成するという意味で全く異なった方式である．このことから，両方式の比較を行い，考察を行うことは非常に重要である．

本稿の目的は，LSIと意味の数学モデルによる意味的連想検索方式を比較し，それぞれの方式の特徴を調べることにある．

§ 1　はじめに

現在，計算機システムによるマルチメディアデータベースは膨大なものになってきている．また，それと同時に，多くの計算機システム群が高速なコンピュータネットワークで接続されている．このような環境においては，膨大なマルチメディアデータ内から的確な情報を獲得する方法が重要である．

的確な情報を獲得する方法としては，ニューラルネットワークモデルによる方式，潜在的意味抽出方式(Latent Semantic Indexing method, LSI[1])及び，我々が提案している意味の数学モデルによる意味的連想検索方式[2], [3]など様々な方式が提案されてきた．これらの方式の中で，本稿ではLSIと意味の数学モデルによる意味的連想検索方式の２つについて比較検討を行う．

LSIでは，まず，検索対象データとそれを説明する語からなるデータ行列Aを得る．次に，Aを特異値分解することによって得られた結果から，空間を作成する．その後，同空間内に検索対象データをベクトル表現する．検索を行う際には，まず，同空間内にユーザからの問い合わせをベクトル表現する．その後，ベクトル表現された問い合わせと各検索対象データとの内積によって，比較を行う．LSIについての詳しい説明は，2節に示す．

一方，意味の数学モデルによる意味的連想検索方式では，まず，単語とそれを説明する基本語からデータ行列Mを得る．次に，M^TMを固有値分解し，得られた固有ベクトルから空間を作成する．その後，検索対象データを空間内にベクトル表現する．検索を行う際には，まず，検索語に応じて動的に空間を選択する．さらに，選択した空間内に検索対象データベクトルを射影する．データ間の比較は，選択した空間内に写像された検索対象データベクトルのノルム値によって行う．意味の数学モデルによる意味的連想検索方式についての詳しい説明は，3節に示す．

以上のように，LSI及び意味の数学モデルによる意味的連想検索方式は，ある空間内でデータをベクトル表現し，比較を行うという方式である．しかしながら，LSIでは静的な空間を用い，意味の数学モデルによる意味的連想検索方式では，文脈に応じて動的な空間を構成するという意味で全く異なった方式である．このことから，両方式の比較を行い，考察を行うことは非常に重要である．

本稿の目的は，LSIと意味の数学モデルによる意味的連想検索方式を比較し，それぞれの方式の特徴を調べることにある．

† 筑波大学大学院理工学研究科，つくば市
Master's Program in Science and Engineering,
University of Tsukuba, Tsukuba, Ibaraki 305-8573, Japan
e-mail：taku@nalab.is.tsukuba.ac.jp

†† 筑波大学大学院システム情報工学研究科，つくば市
Graduate School of Systems and Information Engineering,
University of Tsukuba, Tsukuba, Ibaraki 305-8573, Japan
e-mail：takafumi@nalab.is.tsukuba.ac.jp

††† 筑波大学電子・情報工学系，つくば市
Institute of Information Sciences and Engneering,
University of Tsukuba, Tsukuba, Ibaraki 305-8573, Japan
e-mail：takashi@is.tsukuba.ac.jp

†††† 慶應義塾大学環境情報学部，藤沢市
Faculty of Environmental Information, Keio University,
Fujisawa, Kanagawa 252-8520, Japan
e-mail：kiyoki@sfc.keio.ac.jp

§ 2　潜在的意味抽出方式(LSI)

本節では，LSIで検索を行うまでのプロセスを示す．LSIは，本来ドキュメントの検索方式であるが，ここでは検索対象データをドキュメントに限らないことにする．

2.1　特異値分解

LSIは特異値分解に基づく手法である．2.2節以下の説明をスムーズに行うため，本節では特異値分解の基本性質を示す．

特異値分解とは，rank(A) = r を満たす行列 $A \in \boldsymbol{R}^{m\times n}$ が与えられたとき，

$$A = U\Sigma V^T, \tag{2.1}$$

という3つの行列に分解することである．但し，$U \in \boldsymbol{R}^{m\times n}$ 及び $V \in \boldsymbol{R}^{n\times n}$ は列正規直交行列である．すなわち，Eを単位行列としたとき，

$$U^TU = V^TV = E, \tag{2.2}$$

を満たす行列である．また，$\Sigma \in \boldsymbol{R}^{n\times n}$ は

$$\Sigma = \mathrm{diag}(\sigma_1, \sigma_2, \cdots, \sigma_n), \tag{2.3}$$

となり，

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > \sigma_{r+1} = \sigma_{r+2} = \cdots = \sigma_n = 0, \tag{2.4}$$

を満たす．$\sigma_1, \sigma_2, \cdots, \sigma_n$ をAの特異値という．ここで，

$$U = [\boldsymbol{u}_1\ \boldsymbol{u}_2\ \cdots\ \boldsymbol{u}_n], \tag{2.5}$$

$$V = [\boldsymbol{v}_1\ \boldsymbol{v}_2\ \cdots\ \boldsymbol{v}_n], \tag{2.6}$$

と表す．但し，U及びVの(i, j)要素を，それぞれu_{ij} $(i=1,2,\cdots,m;\ j=1,2,\cdots,n)$及び$v_{ij}$ $(i=1,2,\cdots,n;\ j=1,2,\cdots,n)$としたとき，

$$\boldsymbol{u}_j = (u_{1j}, u_{2j}, \cdots, u_{mj})^T, \tag{2.7}$$

$$\boldsymbol{v}_j = (v_{1j}, v_{2j}, \cdots, v_{nj})^T, \tag{2.8}$$

である．このとき(2.1)は，

$$\mathrm{A} = \sum_{j=1}^{n} \boldsymbol{u}_j \cdot \sigma_j \cdot \boldsymbol{v}_j^{\mathrm{T}} , \tag{2.9}$$

のように書き直せる．(2.9)では n 個の項を加算することで，Aを完全に再現している．しかしながら，Aの特異値は(2.4)の性質を満たす．すなわち，(2.9)の加算は r 回目に近づくほど徐々に無視してよいものになり，(r+1)回目以降の加算は，全て無視できる．したがって，Aは(2.9)における加算の少数の項だけで近似できる．すなわち，A を k 個の項の和で近似した行列を A_k としたとき，

$$\mathrm{A}_k = \sum_{j=1}^{k} \boldsymbol{u}_j \cdot \sigma_j \cdot \boldsymbol{v}_j^{\mathrm{T}} , \tag{2.10}$$

と表すことができる．

LSIでは，(2.10)を用いて検索対象データ及びユーザからの問い合せをベクトル表現し，データ間の比較を行う．次節からは，LSIの具体的な検索手法に入っていく．

2.2 データ行列の作成

本節では，LSIにおけるデータ行列 $\mathrm{A} \in \boldsymbol{R}^{m \times n}$ の作成方法を示す．但し，データ行列とは，列方向に並べたデータと，行方向に並べたデータの関係を表す行列のことである．データ行列Aは，n 個の検索対象データ$(d_1, d_2, \cdots, d_n)$とそのデータを説明する m 個の基本語$(t_1, t_2, \cdots, t_m)$から構成される(Fig. 2.1 参照)．ここで，データ行列 A の(i, j)要素を a_{ij} としたとき，

$$a_{ij} = L(i, j) \times G(i) , \tag{2.11}$$

と表される$(i = 1, 2, \cdots, m; j = 1, 2, \cdots, n)$．但し，$L(i, j)$は d_j 内での t_i の重要度を表しており，$G(i)$ は t_i の全データ内での重要度を表している．すなわち，a_{ij} は d_j 内での t_i のローカルな重みと，全データから見た t_i のグローバルな重みによって決定される．

通常，1つの検索対象データは，少数の基本語によって説明されるため，データ行列Aは疎行列となる．次節では，このデータ行列Aを特異値分解し，分解後のデータをもとに基本語と検索対象データを空間に配置する手法を示す．

2.3 データのベクトル表現

本節では，基本語と検索対象データをベクトル表現する手法を示す．

まず，前節で得られたデータ行列Aを特異値分解する．分解後の結果は，(2.1)に示したように $\mathrm{U}\Sigma\mathrm{V}^{\mathrm{T}}$ の3つの行列となる(Fig. 2.2 参照)．ここで，U及びVをそれぞれ基本語ベクトル及び検索対象データベクトルとする．

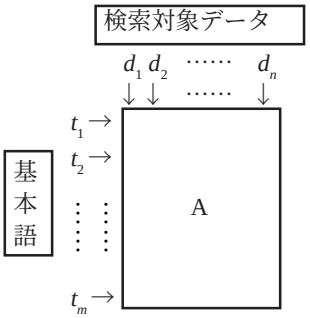

Fig. 2.1. LSIにおけるデータ行列 A.

次に，Aを(2.10)のように，ある k 個の項の和で近似して，A_k を得ることを考える．これはFig. 2.2 において，U, V及びΣの対角線上で斜線を引いている部分に相当する．A_k は元のデータ行列Aを完全には再現していないが，LSIではこの A_k を得ることは重要なことである．なぜなら，A_k を得るということは，基本語と検索対象データを結びつける際に，最も重要な基礎構造を獲得したとも考えられるからである．すなわち，特異値分解によって得られた A_k は，分解前の行列の情報を，低い次元に圧縮していることに相当する．これにより，k次の項までで近似された行列 A_k を得れば，計算回数を減らしながら，検索精度を上げることができるのである．

さて，実際にある k 個の項の和でAを近似し，A_k を得たとする．このとき $\sigma_1, \sigma_2, \cdots, \sigma_k$ は，基本語と検索対象データにかかる重みと考える．すなわち，基本語 t_i $(i = 1, 2, \cdots, m)$ を k 次元空間にベクトル表現したものを $\boldsymbol{t}_{ki}$ とすると，

$$\boldsymbol{t}_{ki} = (\sigma_1 u_{i1}, \sigma_2 u_{i2}, \cdots, \sigma_k u_{ik})^{\mathrm{T}} , \tag{2.12}$$

となる．同様に，検索対象データ d_j $(j = 1, 2, \cdots, n)$を k 次元空間にベクトル表現したものを $\boldsymbol{d}_{kj}$ とすると，

$$\boldsymbol{d}_{kj} = (\sigma_1 v_{j1}, \sigma_2 v_{j2}, \cdots, \sigma_k v_{jk})^{\mathrm{T}} , \tag{2.13}$$

となる．σ_i は(2.4)の性質をもつから，次元が低いほど高い重みがつくことになる．

LSIでは，基本語及び検索対象データをそれぞれ(2.12)及び(2.13)を用いて，空間内にベクトル表現し，さらに，ユーザからの問い合わせもk次元空間にベクトル表現する．その後，問い合わせベクトルを各検索対象データベクトルと比較し，比較結果に応じて検索結果をユーザに返す．次節には，問い合わせベクトルと検索対象データベクトルとの比較を行う方法を示す．

2.4 問い合わせと検索対象データとの比較方法

本節では，まず，ユーザからの問い合わせを k 次元空間にベクトル表現する手法を示す．さらに，問い合せベクトルと検索対象データベクトルとの比較を行う方法を示す．

さて，ユーザからの問い合わせは，幾つかの言葉の集合で与えられるであろう．但し，問い合わせに使用できる言葉は，データ行列Aを作る際に用いた基本語の m 個に含まれているする．このとき，問い合わせを，

$$\boldsymbol{q} = (q_1, q_2, \cdots, q_m)^{\mathrm{T}} , \tag{2.14}$$

で表す．$\boldsymbol{q}$ の各成分 $q_1, q_2, \cdots, q_m$ は，それぞれ $t_1, t_2, \cdots, t_m$ に対応しており，問い合わせに t_i が含まれる場合，q_i = 1

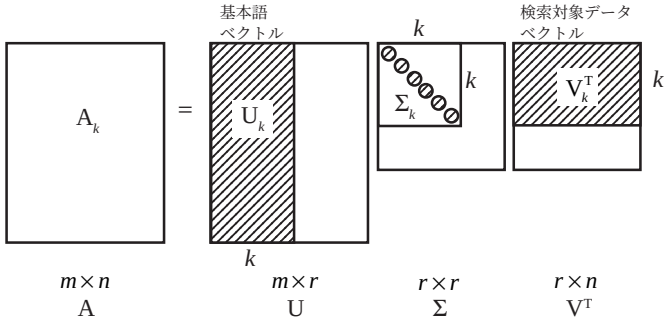

Fig. 2.2. A_k の模式図.

(i = 1, 2, ⋯, m)とする．それ以外の成分は 0 とする．さらに，(2.11)で表したような適切な重みを掛け合わせてもよい．このようにして，ユーザからの問い合わせを $\boldsymbol{q}$ にまとめて表現する．さらに，$\boldsymbol{q}$ を k 次元空間でベクトル表現したものを，

$$\hat{\boldsymbol{q}} = (\hat{q}_1, \hat{q}_2, \cdots, \hat{q}_k)^{\mathrm{T}}, \tag{2.15}$$

とする．$\hat{\boldsymbol{q}}$ は，

$$\hat{\boldsymbol{q}} = \boldsymbol{q}^{\mathrm{T}} \mathrm{U}_k \Sigma_k^{-1}, \tag{2.16}$$

で求められる．すなわち，

$$\hat{q}_i = \frac{1}{\sigma_i} \boldsymbol{q}^{\mathrm{T}} \boldsymbol{u}_i, \qquad (i = 1, 2, \cdots, k), \tag{2.17}$$

である．(2.17)における $1/\sigma_i$ は第 i 次元における重みである．

(2.16)を用いて問い合わせをk次元空間にベクトル表現することで，すでに k 次元空間にベクトル表現してある検索対象データとの比較ができる．問い合わせベクトルと，検索対象データベクトルとの類似度を測るには，ある共通の尺度が必要となる．ここでは，共通の尺度として，問い合わせベクトルと検索対象データベクトルとの余弦(cos)の値を用いる．すなわち，i番目の検索対象データとの比較をする際には，

$$\cos\theta_{ki} = \frac{(\hat{\boldsymbol{q}}, \boldsymbol{d}_{ki})}{|\hat{\boldsymbol{q}}||\boldsymbol{d}_{ki}|}, \tag{2.18}$$

を計算し，$\cos\theta_{ki}$ の値が大きい順に類似度が高いとみなす．但し，θ_{ki} はk次元空間において，$\hat{\boldsymbol{q}}$ と $\boldsymbol{d}_{ki}$ がなす角である．検索結果は，$\cos\theta_{ki}$ の値がある閾値を上回った検索対象データを，値が大きい順に返すものとする．

以上が LSI における検索までの流れである．次節からは，意味の数学モデルによる意味的連想検索について，詳しく説明していく．

§ 3　意味の数学モデルによる意味的連想検索方式

本節では，意味の数学モデルによる意味的連想検索方式について説明していく．以下では，意味の数学モデルによる意味的連想検索方式のことを，単に意味モデルと呼ぶことにする．

3.1　データ行列の作成

本節では，意味モデルにおけるデータ行列 $\mathrm{M} \in \boldsymbol{R}^{m \times n}$ の作成方法を示す．データ行列Mは，単語とそれを説明する基本語の関係を表す行列である．このデータ行列Mは，m 個のn次元行ベクトルを列挙したものと考える．すなわち，データ行列Mは m 個の単語 $\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_m$ のそれぞれを，n 個の基本語 $f_1, f_2, \cdots, f_n$ の中の幾つかを用いて説明し，それらを列挙したものと考える(Fig. 3.1 参照)．

さて，データ行列Mの作成方法を具体的に示すために，第 i 番目の単語を，

$$\boldsymbol{w}_i = (w_{i1}, w_{i2}, \cdots, w_{in}), \quad (i = 1, 2, \cdots, m), \tag{3.1}$$

と定義する．$\boldsymbol{w}_i$ の各成分 $w_{i1}, w_{i2}, \cdots, w_{in}$ は，それぞれ $f_1, f_2, \cdots, f_n$ に対応しており，$\boldsymbol{w}_i$ の説明に f_j が使用されている場合，w_{ij} に0でない値を定める．また，$\boldsymbol{w}_i$ の説明に使用されていない f_j に対応する成分は，全て0とする．このようにして，全ての単語についてベクトルの成分を決定し，$\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_m$ の順に，それぞれデータ行列Mの第1行，第2行，…，第 m 行とする．

以上の手順でデータ行列Mが作成できる．次節では，データ行列Mからメタデータ空間S_{M}を作成する手順を示す．

3.2　メタデータ空間の作成

意味モデルの特徴は，検索者からの問い合わせに応じて，動的に空間を選択することである．本節では，動的に空間を選択する際に，そのもとの空間となるメタデータ空間 S_{M} の作成方法を示す．

まず，データ行列Mの相関行列 $\mathrm{M}^{\mathrm{T}}\mathrm{M}$ を計算する．これは $\mathrm{M}^{\mathrm{T}}\mathrm{M}$ を計算することで，f_i と f_j の関係を調べていることに相当する(i = 1, 2, ⋯, n; j = 1, 2, ⋯, n).

次に，$\mathrm{M}^{\mathrm{T}}\mathrm{M}$ を固有値分解して，

$$\mathrm{M}^{\mathrm{T}}\mathrm{M} = \mathrm{Q}\Lambda\mathrm{Q}^{\mathrm{T}}, \tag{3.2}$$

とする．但し，$\Lambda \in \boldsymbol{R}^{n \times n}$ は，

$$\Lambda = \mathrm{diag}(\lambda_1, \lambda_2, \cdots, \lambda_\nu, 0, \cdots, 0), \qquad (0 \le \nu \le n), \tag{3.3}$$

であり，

$$\nu = \mathrm{rank}(\mathrm{M}^{\mathrm{T}}\mathrm{M}), \tag{3.4}$$

となる．また，$\mathrm{Q} \in \boldsymbol{R}^{n \times n}$ は，

$$\mathrm{Q} = [\boldsymbol{q}_1\ \boldsymbol{q}_2\ \cdots\ \boldsymbol{q}_n], \tag{3.5}$$

であり，$\boldsymbol{q}_i (i = 1, 2, \cdots, n)$は $\mathrm{M}^{\mathrm{T}}\mathrm{M}$ の正規化された固有ベクトルである．$\mathrm{M}^{\mathrm{T}}\mathrm{M}$は対称行列であるため，固有値は全て実数であり，対応する固有ベクトルは互いに直交している．$\mathrm{M}^{\mathrm{T}}\mathrm{M}$ の固有ベクトルを用いて，メタデータ空間 S_{M} を次のように定義する．

$$S_{\mathrm{M}} := \mathrm{span}(\boldsymbol{q}_1, \boldsymbol{q}_2, \cdots, \boldsymbol{q}_\nu). \tag{3.6}$$

(3.6)において，$\{\boldsymbol{q}_1, \boldsymbol{q}_2, \cdots, \boldsymbol{q}_\nu\}$ は S_{M} の正規直交基底である．

ここで，S_{M}から固有部分空間への射影の集合Π_νを考える．その際，λ_iに対応する固有部分空間への射影 P_{λ_i} を次のように定義する．

$$P_{\lambda_i} : S_{\mathrm{M}} \to \mathrm{span}(\boldsymbol{q}_i), \tag{3.7}$$

このとき，Π_ν を，

$$\begin{aligned}\Pi_\nu = \{\ & 0, P_{\lambda_1}, P_{\lambda_2}, \cdots, P_{\lambda_\nu}, \\ & P_{\lambda_1} + P_{\lambda_2}, P_{\lambda_1} + P_{\lambda_3}, \cdots, P_{\lambda_{\nu-1}} + P_{\lambda_\nu}, \\ & \quad \vdots \\ & P_{\lambda_1} + P_{\lambda_2} + \cdots + P_{\lambda_\nu}\},\end{aligned} \tag{3.8}$$

と定義する．k 次元の固有部分空間は ${}_\nu C_k (k = 1, 2, \cdots, \nu)$個存在するから，射影の総数は 2^ν となる．すなわち，意味モデルは 2^ν 通りの意味の表現能力を持つ．

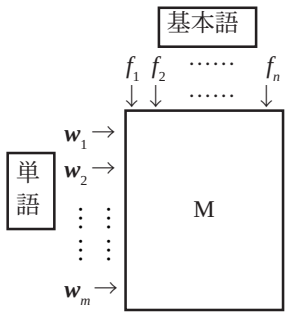

Fig. 3.1. 意味モデルにおけるデータ行列M.

3.3　検索対象データベクトルの作成

本節では，検索の対象となるデータをベクトル表現する手法の１つを示す．

まず，各検索対称データごとに特徴付けを行う．各検索対象データの特徴付けとは，各データごとに$\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_m$の中から$t$個の単語を選ぶことに相当する．但し，$t$個の単語は，各検索対象データの印象を強く表す単語を選ぶことにする．また，tの値は各検索対象データごとにランダムな値でよい．ここで，ある検索対象データpで選んだt個の単語が，$\boldsymbol{o}_1, \boldsymbol{o}_2, \cdots, \boldsymbol{o}_t$であるとする．但し，$\boldsymbol{o}_1, \boldsymbol{o}_2, \cdots, \boldsymbol{o}_t$のそれぞれは，$\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_m$の中のいずれかに一致する行ベクトルで，

$$\boldsymbol{o}_i = (o_{i1}, o_{i2}, \cdots, o_{in}), \qquad (i = 1, 2, \cdots, t), \tag{3.9}$$

と表される．このとき，検索対象データpをベクトル表現したものを$\boldsymbol{p}$とすると，

$$\boldsymbol{p} = \left(\mathrm{sign}(o_{\ell_1 1}) \max_{1\le i\le t} |o_{i1}|,\ \mathrm{sign}(o_{\ell_2 2}) \max_{1\le i\le t} |o_{i2}|,\right.$$
$$\left. \cdots,\ \mathrm{sign}(o_{\ell_n n}) \max_{1\le i\le t} |o_{in}|\right)^{\mathrm{T}}, \tag{3.10}$$

として各成分を計算する．但し，$\mathrm{sign}(o_{\ell_k k})$ $(k=1,2,\cdots,n)$は，$o_{\ell_k k}$の符号(+または–)を表す．また，ℓ_k $(k=1,2,\cdots,n)$は，

$$|o_{\ell_k k}| = \max_{1\le i\le t} |o_{ik}|, \tag{3.11}$$

を満たすように決定する．

3.4　動的な空間選択

本節では，意味モデルの特徴である，問い合わせに応じた動的な空間選択について説明する．

ユーザからの問い合わせは，幾つかの単語の集合で与えられるであろう．但し，問い合わせに使用できる単語は，$\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_m$に含まれているとする．このとき，ユーザからの問い合わせの単語の集合を，

$$s_\ell = \{\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_\ell\}, \tag{3.12}$$

とする．但し，ℓはユーザが問い合わせに使用した単語の個数であり，$\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_\ell$のそれぞれは，$\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_m$のいずれかに一致する．

(3.12)のようにs_ℓが与えられたとき，問い合わせに応じた動的な部分空間の選択は，以下の手順によって実現される．

Step 1:　$\boldsymbol{u}_i (i = 1, 2, \cdots, \ell)$をメタデータ空間$S_{\mathrm{M}}$にフーリエ展開する．

メタデータ空間S_{M}を構成する$\{\boldsymbol{q}_1, \boldsymbol{q}_2, \cdots, \boldsymbol{q}_v\}$は，正規直交基底であるから，$\boldsymbol{u}_i (i = 1, 2, \cdots, \ell)$を$S_{\mathrm{M}}$にフーリエ展開できる．すなわち，メタデータ空間$S_{\mathrm{M}}$への$\boldsymbol{u}_i$のフーリエ展開とは，

$$\boldsymbol{u}_i = \sum_{j=1}^{v} (\boldsymbol{u}_i, \boldsymbol{q}_j)\boldsymbol{q}_j, \tag{3.13}$$

として，メタデータ空間S_{M}に$\boldsymbol{u}_i$をマッピングすることを意味する．

Step 2:　s_ℓの重心$\boldsymbol{g}^+(s_\ell)$を計算する．

まず，u_{ij}を次のように定義する．

$$u_{ij} := (\boldsymbol{u}_i, \boldsymbol{q}_j), \qquad (j = 1, 2, \cdots, v). \tag{3.14}$$

さらに，$\boldsymbol{u}^*$を，

$$\boldsymbol{u}^* := \left(\sum_{i=1}^{\ell} u_{i1}, \sum_{i=1}^{\ell} u_{i2}, \cdots, \sum_{i=1}^{\ell} u_{iv}\right)^{\mathrm{T}}, \tag{3.15}$$

と定義する．このとき，メタデータ空間S_{M}におけるs_ℓの重心$\boldsymbol{g}^+(s_\ell)$を，

$$\boldsymbol{g}^+(s_\ell) := \frac{\boldsymbol{u}^*}{\|\boldsymbol{u}^*\|_\infty}, \tag{3.16}$$

と定義する．但し，$\|\cdot\|_\infty$は無限大ノルムである．

Step 3:　問い合わせs_ℓの重心$\boldsymbol{g}^+(s_\ell)$に応じて射影$P_{\varepsilon_s}(s_\ell)$を決定し，部分空間の選択を行う．

射影$P_{\varepsilon_s}(s_\ell)$は，ある閾値$\varepsilon_s (0 < \varepsilon_s < 1)$が与えられたとき，

$$P_{\varepsilon_s}(s_\ell) := \sum_{j\in\Lambda_{\varepsilon_s}} P_{\lambda_j} \in \Pi_v, \tag{3.17}$$

(但し，$\Lambda_{\varepsilon_s} := \{j \mid |g_j| > \varepsilon_s\}$),

と定義される．但し，g_jは$\boldsymbol{g}^+(s_\ell)$の第j成分である．$P_{\varepsilon_s}(s_\ell)$により，選ぶべき部分空間が決定される．

3.5　問い合わせとの相関の計量方法

ここでは，前節と同様に問い合わせとしてs_ℓが与えられたときに，s_ℓと検索対象データとの比較をどのように行うかを説明する．

まずは，s_ℓに応じて選択した部分空間に，検索対象データベクトルを写像する．この写像により，各検索対象データベクトルは，部分空間内での大きさをもつ．部分空間はs_ℓに応じて選択されているから，この空間内で大きいベクトルほど，問い合わせとの相関が強いデータといえる．すなわち，問い合わせと各検索対象データとの相関の計量は，選択した部分空間における検索対象データベクトルのノルムによって行う．ここで，問い合わせとしてs_ℓが与えられたときの，検索対象データ$\boldsymbol{x}$の部分空間内でのノルム$\rho(\boldsymbol{x}; s_\ell)$を，以下のように定義する．

$$\rho(\boldsymbol{x}; s_\ell) := \frac{\sqrt{\sum_{j\in\Lambda_{\varepsilon_s}\cap S} (g_j x_j)^2}}{\|\boldsymbol{x}\|_2}, \tag{3.18}$$

(但し，$S = \{i \mid \mathrm{sign}(g_i) = \mathrm{sign}(x_i)\}$，$j \in \Lambda_{\varepsilon_s}$).

検索結果は，$\rho(\boldsymbol{x}; s_\ell)$の値が大きい順にユーザに返される．

§４　LSIと意味モデルの特徴

本節では，２節及び３節での説明をもとに，LSIと意味モデルの特徴を，比較を行いながら探る．LSIと意味モデルでは，基本コンセプトが特異値分解であるか，固有値分解であるかという違いがあるが，ここではそれ以外を比較し，大きく異なる部分をあげていく．

4.1　データ行列に関連した特徴

LSIは，もともとドキュメント検索の手法として提案されたものである．そのため，検索対称をドキュメントとしたとき，他の正規直交座標系を用いる検索手法に比べて，データ行列Aが非常に作りやすくなっている．なぜなら，ドキュメント内に存在する単語を，そのまま基本語に割り

当てることができるからである．また，各単語の出現頻度に応じてtf・idf法[4]を用いれば，Aの各要素は簡単に決定することができる(tf・idf法についてはAppendix A参照)．これは検索対称データが多くなっても，簡単に，しかもほとんど自動的にデータ行列が作れるということであり，意味モデルと比べてLSIの非常に優れている点である．

一方，意味モデルのデータ行列は，作成に非常に手間がかかる．しかしながら，一度データ行列Mを作り，M^TMの固有値分解によってv個の固有ベクトルが得られれば，非常に有効な検索空間が完成する．なぜなら，検索対象データが増加してもデータ行列を作り直す必要はなく，同じ空間内に新たにベクトル表現するだけで検索対象として加えることができるからである．さらに，このデータ行列は，データの種類(ドキュメント，音楽，絵画等)に応じて作り直すことなく，同じデータ行列から作成した空間内でデータの比較を行うことができる．これはLSIと比較して，意味モデルが非常に優れている点である．

4.2 検索空間に関連した特徴

LSIでデータをベクトル表現する座標系は，直交座標系ならばどのようなものでもかまわない．例えば，k次元のデカルト座標系等を使用する(但し，$k \le r$)．LSIは特異値分解によって，低い次元ほど高い重みをつけて，もとのデータ行列Aの情報を圧縮する．そのため，第k次元までの計算で，もとの行列Aの情報を精度良く再現し，少ない計算量で検索が実現できる．これはLSIの非常に優れた特徴である．

計算する次元数を増加させていくと，Aをさらに精度良く再現できるため，当然検索精度も上がるはずである．しかしながら，計算する次元数を上げるということは，検索空間のサイズも同時に上がることを意味する．検索空間のサイズが上がるということは，データ間の比較の際に，検索結果を悪化させる要因を含む可能性も上がるということである．LSIはk次元空間での検索において，全空間(全次元の情報)でデータ間の比較を行うため，不要な情報が混在しても，それを排除することはできない．そのため，検索結果の次元数kへの依存性を調べることは，LSIにおいて非常に重要なことである．

一方，意味モデルは，M^TMの固有値分解によって得られた，非零の固有値に対応したv個の固有ベクトルでv次元正規直交空間を生成し，この空間内でデータ間の比較を行う．比較を行う際に，意味モデルはこの空間内で問い合わせに応じて部分空間を選択する．部分空間は，ユーザからの問い合わせに応じて動的に選択し，不要な次元の情報は排除する．これは，意味モデルの非常に優れた特徴である．意味モデルは，動的な空間選択により，検索に有用な情報だけを残すため，非常に良い条件での検索を実現している．

§5 実験

本節では，ドキュメントを検索対象として，LSIと意味モデルの検索結果を比較する．

意味モデルにおけるデータ行列Mの作成をする際には，“Longman Dictionary of Contemporary English[5]”(以下，LD)を使用した．LDは，約2,000語の基本語だけを用いて，約56,000語の見出し語を説明している英英辞典である．ここでは，LDの約2,000語の基本語をデータ行列Mのn個の基本語($f_1, f_2, \cdots, f_n$)として扱った．さらに，$f_1, f_2, \cdots, f_n$を用いることによって，i番目の単語$\boldsymbol{w}_i$を説明した($i = 1, 2, \cdots, m$)．$\boldsymbol{w}_i$は(3.1)に定義したように，$w_{i1}, w_{i2}, \cdots, w_{in}$で構成され，これらはそれぞれ，$f_1, f_2, \cdots, f_n$に対応している．ここでいう説明とは，$\boldsymbol{w}_i$の説明に$f_j$が使われているとき，

$$w_{ij} = \begin{cases} 0 & (f_j\text{が説明に使用されないとき}) \\ 1 & \begin{pmatrix} f_j\text{が肯定の意味で使われているとき} \\ f_j\text{自身が見出し語であるとき} \end{pmatrix} \\ -1 & (f_j\text{が否定の意味で使われているとき}) \end{cases}$$

として，w_{ij}を決定することである．このようにして，全てのw_{ij}を説明し，データ行列Mを作成した．

LSIにおけるデータ行列Aは，4.1節に示したLSIの特徴をいかしつつ，tf・idf法を用いることによって自動的に作成した．作成した行列のサイズは，1889×50となった．

検索対象となるドキュメントは，50個用意した．これらは，5つのデータごとに10個のカテゴリに属する．10個のカテゴリ名は，それぞれ2つの英単語からなる(Table 5.1参照)．各ドキュメントは，それぞれが属するカテゴリについて書かれた英文書である．各実験において検索を行う際には，カテゴリ1～10のいずれかのカテゴリ名で検索を行い，正解は各カテゴリに属する5つのドキュメントであるとする．

5.1 LSIにおける検索結果の次元数依存性

本節では，LSIにおいてデータ間の比較を行った際の，検索結果の次元数依存性について調べる．特異値分解によって得られた特異値は全て非零であったため，最大50次元までの空間で検索を行うことができる．

実際に検索を行うにあたり，検索語としては，“car, drive”，“computer, network”，“gold, diamond”，“nature, plant”及び“star, space”を与えた．また，検索結果の上位5位以内に入っ

Table 5.1. 10個のカテゴリ．

No.	カテゴリ名	No.	カテゴリ名
1	car, drive	6	image, recognition
2	computer, network	7	nature, plant
3	food, eat	8	river, fish
4	gold, diamond	9	sport, ball
5	human, brain	10	star, space

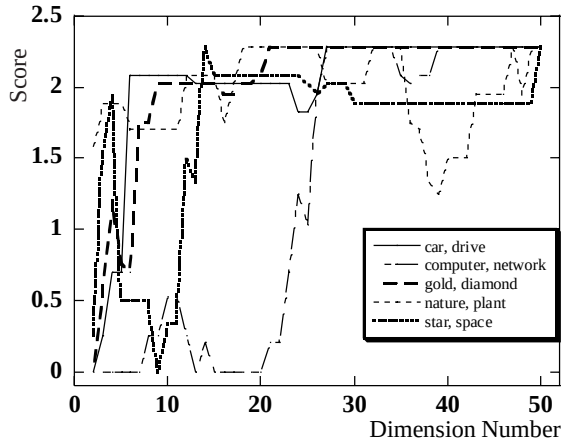

Fig. 5.1. LSI における検索結果の次元数依存性.

ている正解数と順位に応じて, 各次元ごとにスコアを計算し, グラフ化した. その実験結果を Fig. 5.1 に示す. 但し, 第 k 次元におけるスコア x_k は, 次の式で計算した.

$$x_k = \begin{cases} \sum_{i=1}^{N_k} \frac{1}{r_{ki}} & (N_k = 1,2,\cdots,5), \\ 0 & (N_k = 0). \end{cases} \tag{5.1}$$

(5.1)における N_k は第 k 次元における 5 位以内の正解数であり, r_{ki} は第 k 次元における正解データの順位である. (5.1)は, 検索結果の順位まで考慮したスコアの決め方である. すなわち, これは各次元における検索結果の良し悪しを決める尺度として, 適している.

Fig. 5.1 から, 低い次元(10 次元以下)では検索結果にかなりのばらつきがあることがわかる. しかしながら, 問い合わせに“ computer, network ”を使用したとき以外は, 約 15 次元(最大次元数の約 30%)までの計算で, 充分な検索結果が得られており, 4.2 節で示した LSI の優れた特徴が確認できる. また, 問い合わせに“ computer, network ”を使用したときは, 27次元程度でほぼ安定した結果が得られているものの, それ以前の結果では, 満足な結果が得られていない. これは, “ computer, network ”と比較的関係の深い, “ image, recognition ”に関するドキュメントが, 低い次元数のときに上位に検索されたことが原因と思われる.

全体として見れば, 次元数の増加に伴って検索結果が良くなり, 30 次元前後からは安定した結果となっている. しかしながら, 問い合わせに“ nature, plant ”を使用した場合に高い次元で検索結果が多少悪くなっているように, 次元を上げていった場合, 検索結果を悪化させる要因を含んで検索をしてしまう可能性もある. 但し, 最大次元で検索を行った場合, 全ての場合でスコアが最大となっていることから, 最大次元まで計算を行うことで, 検索結果を悪化させる要因を緩和することができているといえる. しかしながら, LSI において最大次元で検索を行うことは実用的ではないため, ほとんどの場合, 不要な情報を少し含んでしまうであろう.

以上から, LSI で検索を行う際, 最大次元の約 30% の次元数で検索を行うと, ほとんどの場合, 非常に良い結果が得られるといえる. また, 最大次元の約 60% の次元数で検索を行うと, 例外的な問い合わせに対しても, 充分な結果を得ることができる. LSI では, データの比較を行う空間の次元数が, その性能に大きく影響を与えるが, 本節の実験で次元数を決めるだいたいの目安がわかった. 次節では, ここで得られた次元数を決める目安をもとに次元数を固定し, LSI と意味モデルとの比較を行うことにする.

5.2 再現率と適合率による比較

ここでは, 前節の結果から, LSI で比較を行う空間を 15 次元と 30 次元に固定し, LSI と意味モデルとの検索結果の比較を行う. 本節での実験は LSI 及び意味モデルのパラメタが大きく違うため, Table 5.2 にその違いをまとめて

Table 5.2. 5.2 節の実験における LSI 及び意味モデルのパラメタの違い.

LSI		意味モデル
Aは, ドキュメントと単語の関係を直接表す行列である. すなわち, 50 の各ドキュメントを, 1889 の単語で直接説明している. Aの(i, j)要素は, 各ドキュメントごとに単語の出現頻度に応じてtf･idf法により決定した.	ドキュメント, n, 単語, m, A, $m = 1889$, $n = 50$.	Mは, 基本語と単語の関係を表す行列である. ドキュメントの特徴付けは, 単語とその重みの組を複数与えることで行った. 但し, 単語の重みは, 各ドキュメントごとに単語の出現頻度に応じて tf･idf 法により決定した.
	データ行列	基本語, n, 単語, m, M, $m = 2328$, $n = 2328$.
15 次元及び 30 次元に固定 (5.1 節の実験より決定)	検索を行う空間	ユーザからの問い合わせに応じて動的に選択
$\cos\theta_{ki} = \frac{(\hat{\boldsymbol{q}}, \boldsymbol{d}_{ki})}{\lvert\hat{\boldsymbol{q}}\rvert\lvert\boldsymbol{d}_{ki}\rvert}$, (2.18)	類似度の計量式	$\rho(\boldsymbol{x}; s_\ell) := \frac{\sqrt{\sum_{j \in \Lambda_{\varepsilon_s} \cap S} (g_j x_j)^2}}{\lVert \boldsymbol{x} \rVert_2}$, (3.18)
	その他のパラメタ	$\varepsilon_s = 0.15$

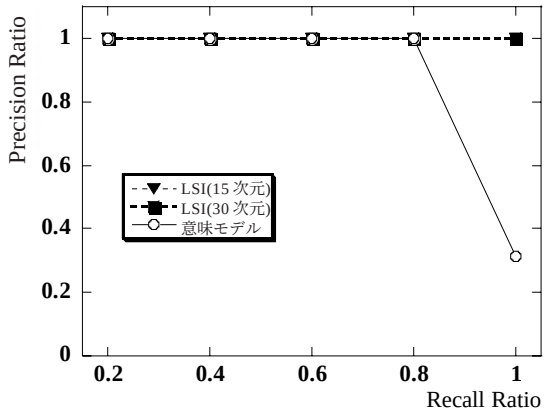

Fig. 5.2. human, brainで検索したときの再現率–適合率グラフ.

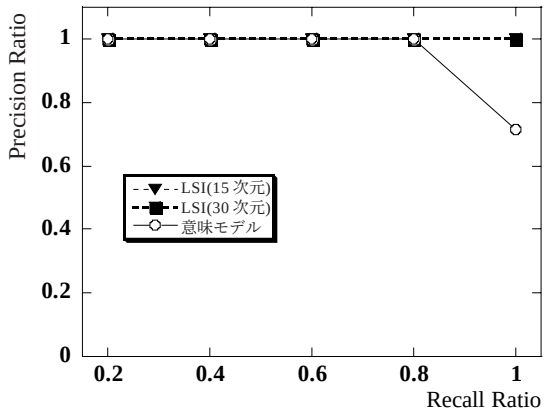

Fig. 5.3. food, eatで検索したときの再現率–適合率グラフ.

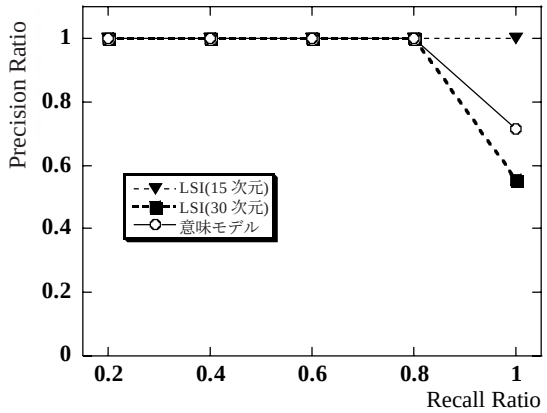

Fig. 5.4. river, fishで検索したときの再現率–適合率グラフ.

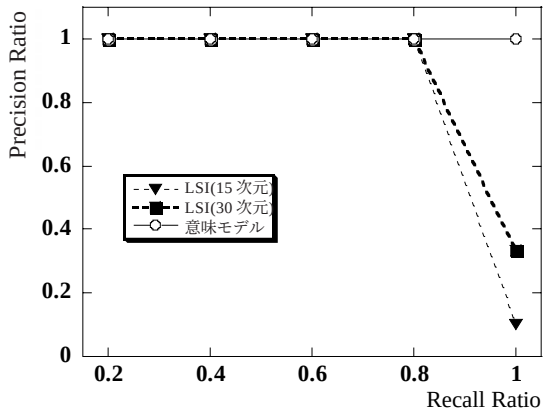

Fig. 5.5 . sport, ballで検索したときの再現率–適合率グラフ.

おいた．また，検索結果の比較は，再現率と適合率によって行った(再現率及び適合率については，Appendix B 参照)．

実際に比較を行うにあたり，問い合わせとしては，“ human, brain ”, “ food, eat ”, “ river, fish ” 及び “ sport, ball ” をそれぞれ与えた．実験結果をそれぞれFig. 5.2, Fig. 5.3, Fig. 5.4 及びFig. 5.5 に示す．Fig. 5.2 及びFig. 5.3 では，LSI の結果が 15 次元及び 30 次元で一致し，意味モデルよりも良い結果を出している．一方，Fig. 5.5 では , 意味モデルがLSI の 15 次元及び 30 次元よりも良い結果を出している．また，Fig. 5.4では, LSIの15次元で比較を行った場合が最も良く，続いて，意味モデル, LSIの30次元という順番になった．しかしながら, どの問い合わせにおいても, LSI及び意味モデルの両方とも良い検索結果が得られており，その差はわずかである．このことから，ドキュメント検索において，両方式共に非常に優れた検索方式であるといえる．また, LSI の 15 次元と 30 次元の結果がほぼ変わらないことから，前節にあげた“ computer, network ”のような例外を除けば，ほとんどの場合，最大次元数に対して約30%の次元数の空間における検索で，充分な結果が得られるといえる．但し，LSIと意味モデルは共にパラメタに依存する部分もあるため，適切なパラメタ設定は必須である．その意味で，今回，LSI のデータ行列 A の(i, j)要素の決定と，意味モデルにおけるデータの特徴づけの際に用いた tf・idf 法は，LSI と意味モデルにおいて，非常に良いパラメタを与える有効な手法であるといえる．

§6　結論と今後の課題

本稿では, LSIと意味モデルの比較を行い, それぞれの特徴を調べた．結論は以下の通りである．

(1) LSIは最大次元まで計算を行うことで, 検索結果を悪化させる要因を緩和することができる．
(2) LSI 及び意味モデルは，ドキュメント検索において，非常に良い結果を返す優れた方式である．
(3) LSI は最大次元数に対して約 30% の次元数の空間で検索を行えば, ほとんどの場合, 充分な検索結果を返す．
(4) LSI と意味モデルは，結果がパラメタに依存する部分もあるが，tf・idf 法は，LSI のデータ行列作成及び意味モデルにおけるデータの特徴付けの際に，非常に良いパラメタを与える有効な手法である．

今後の課題としては，さらに多くのデータ数で実験を行うことがまず上げられる．また，ドキュメント以外のメディアで実験を行うことや，今回とは違った視点での比較を行うことも必要である．さらに, LSIにおいては，意味モデルのように動的に空間を選択する手法を考えることや，意味モデルにおいて，ε_s などのパラメタに関する考察をすることも重要である．

また, 他の課題として, LSI及び意味モデルの基本コンセプトとなっている，特異値分解及び固有値分解は，行列のサイズが大きくなった際に，計算時間の問題が生じる．

特異値分解及び固有値分解は，両方ともかなり重い処理であり，データ数が増加した場合，非常に長い計算時間を必要とする．これを回避するために，特異値分解及び固有値分解の計算時間縮小のためのアルゴリズムを考えること，または特異値分解及び固有値分解に変わる，新たな基本コンセプトの導入なども今後の課題として上げられる．

参考文献

[1] MICHAEL W. BERRY, SUSAN T. DUMAIS and GAVIN W. O 'BRIEN,“ SING LINEAR ALGEBRA FORINTELLIGENT INFORMATION RETRIEVAL,” Sociaty for Industrial and Applied Mathematics, Vol. 37, No. 4, pp.573-595, December 1995.

[2] Kitagawa, T., Kiyoki, Y.,“ A mathematical model of meaning and its application to multi-database systems,” Proc. IEEE Int. Workshop on RIDE Interoperability in Multidatabase Systems, pp.130-135, April 1993.

[3] Takashi Kitagawa, Yasushi Kiyoki and Kyoko Nakamura,“ A Semantic Media Data Search Method Based on a Mathematical Model of Meaning for Multimedia Information Systems,” Information Modelling and Knowledge Bases, H. Jaakkola et al. (Eds.), IOS Press, 1999.

[4] 馬場肇,“ 日本語全文検索システムの構築と活用，”ソフトバンクパブリッシング，1998.

[5] Longman Dictionary of Contemporary English, Longman, 1987.

Appendix A　tf・idf 法

ここでは, 文書群における単語の重み付けとして, 非常に効果的な tf・idf 法について述べる．tf・idf 法とは,「ある文書群における高頻度語の重要度を下げ, 低頻度語の重要度を上げる」という考え方の重み付け法である．この手法の具体的な説明をする前に，式中に用いるデータの定義をしておく．

まず, 検索対象となる全文書が N 個あるとし，その中の i 番目のデータを $D_i(i=1,2,\cdots,N)$ とする．また，D_i 中の全単語の数を M_i とし，D_i の j 番目の単語を w_{ij}，その出現頻度を x_{ij} とする(但し，$j=1,2,\cdots,M_i$). さらに，全文書中で w_{ij} を含む文書数を n_{ij} とする．

以上のデータを用いて，w_{ij} の重み x'_{ij} は，

$$x'_{ij} = x_{ij} \log\left(\frac{N}{n_{ij}}\right), \tag{A.1}$$

と計算される．この方式では，N 個の文書中で，w_{ij} が多くのデータ中に現れる普遍的な単語である場合に x'_{ij} は小さくなる．逆に，w_{ij} が特定の文書中にしか現れない場合には x'_{ij} が大きくなる. すなわち, 各文書を強く特徴付けるであろう単語ほど，強い重み付けがされることになる．

Appendix B　再現率と適合率

ここでは，再現率及び適合率について説明する．再現率とは，検索条件に適合した情報がどれくらいの率で検索されたかを測るための指標である．また，適合率とは，検索条件に適合した情報がどれくらいの率で検索されたかを測るための指標である．具体的には，次のようになる．

$$\text{再現率} = \frac{\text{システムの検索結果に含まれる正解数}}{\text{検索対象データに含まれる本来の正解数}},$$

$$\text{適合率} = \frac{\text{システムの検索結果に含まれる正解数}}{\text{システムの検索結果出力数}}.$$

理想的には，再現率及び適合率の両方が 1 に近づくのが望ましい．しかしながら，実際には，再現率と適合率の間には，再現率が上がると適合率が下がり，再現率が下がると適合率が上がるというような関係がある．