【論文】

自己組織化マップによって分類された単語に基づく文書検索

井上 正人[*1]

Document Search Using Words Categorized by Self-Organizing Maps

Masato Inoue

Abstract

We propose a document search method based on similarity of documents. Using Japanese Linux FAQ documents, Japanese documents are split into words. Words are categorized by Self-Organizing Maps. Documents are expressed as histograms of categorized words. Distances between documents are defined and these distances express similarity of documents.

Key words: Self-Organizing Maps, Word Vector, Document Search

1 はじめに

インターネット上には膨大な情報があり，日々，新しい情報が加わっている．これらの情報を有効に利用するためには，効率的に欲しい情報を探し出すことが必要で，Google[1]などの検索サイトは欠かせないものである．検索サイトではウェブ上の情報を収集し，解析して，インデックスを作成し，ブラウザを用いてキーワード検索ができるようなシステムになっている．情報が日々更新されるよう，常に巡回ロボットがウェブサイトを訪れて，新しい情報を収集している．

コンピュータ内の情報を探すシステムとしてはGoogle デスクトップや全文検索システムNamazu[2]などがある．これらも，コンピュータ内の情報を解析して，データベース化しており，簡単な操作で，ソフトウェアの検索ボックスやブラウザの画面でキーワードを入力することにより情報を検索できるものである．

キーワードを入力して情報を探す方法は一般に広く用いられているが，キーワードに関するある程度の知識がないと情報をうまく探せない．Kohonenたちのグループは，自己組織化マップ(Self-Organizing Maps SOM)[3]を用いて大量の文書を分類するWEBSOM[4,5]の方法を提唱した．2次元の格子上に配置されたユニット上に文書が分類されるが，内容の近い文書が近くのユニットに配置されるようになっており，同一ユニットには非常に内容の近い文書が集まる．ハイパーリンク等を用いて，文書の内容などが表示されるようになっている．文書の分類は，文書を単語に分解し，単語をSOM で分類して，文書を分類された単語のヒストグラムで表す．それらを，もう一度SOMで分類する．

文書クラスタリングについてはその他さまざまな方法が提唱されている[6]．

著者らは，これまでにWEBSOMの方法を用いて，2種類の日本語で書かれたメールマガジンを分類することを試みた[7]．

本論文では，文書を分類するのではなく，文書の集まりの中から，一つの文書を選択したとき，それに近い文書を選択するシステムを作成する．対象は日本語の文書で，文書を単語に分解して単語をSOM で分類し，文書を分類された単語に基づいてヒストグラムで表した後，文書間の距離を計算し，文書を入力すると，最も近い距離の文書が表示される．

扱う文書としては，文書の集まりとして統一性があり，かつ各文書が独立して記述されているLinux

*1 海上保安大学校　inoue@jcga.ac.jp

関連の日本語の文書の集まりである Linux JF (Japanese FAQ)[8] Project のドキュメントを用いた．文書はカーネルについて書かれた Kernel-HOWTO.txt やモデムについて書かれた Modem-HOWTO.txt など細かく分かれており．各文書は独立している．

2 処理の方法

処理の大まかな手順は次のようになる．

1) 文書の前処理
2) 分かち書きによる単語の抽出
3) 単語ベクトルの割り当て
4) SOM による単語の分類
5) 単語の分類に基づく文書のヒストグラム
6) 文書間の距離の計算

それぞれの手順について詳しく述べる．

2.1 前処理

文書の数は 384 である．文字コードは Shift-JIS なので OS の文字コード UTF-8 に変換して作業する．元の文書には文字化けして読めない文書があったので 8)のウェブサイトから対応する文書をダウンロードした．SSL の証明書のような無意味な英数字からなる非常に長い行があるのでこれらの行は削除する．後の処理がしやすいよう，全角英数字を半角英数に，アルファベットはすべて小文字にし，特殊文字は空白に置き換える．各行の終わりにある余分な改行は削除する．これらの作業は Perl で行った．

2.2 分かち書きによる単語の抽出

前処理の終わった文書の分かち書きを行う．処理には形態素解析ソフト MeCab[9]を用いた．MeCab の処理には辞書が必要である．辞書は Mecab との組み合わせでよく用いられる IPA (Information-technology Promotion Agency, Japan) の辞書を用いた．ソフトと辞書は Fedora 用に作成されているパッケージをインストールして使用した．分かち書きの後，句読点などを除去し各文書から単語を取り出した．単語総数は 59846 個である．あまり出現頻度低いと 2.3 節の前と後ろの文脈を表わす平均がばらつくので出現頻度が 20 回以内の単語を除く．また，頻度の非常に多い単語は，各文書に均等に現れる場合，文書の違いを表わさず，偏った場合，違いを極端に際立たせるので出現頻度が 2000 回以上の単語を除く．これらを除いた 7969 個の単語を基に文書のヒストグラムを作成する．

2.3 単語ベクトルの作成

WEBSOM の方法を用いて各単語にランダムなベクトルを割り当てる．ベクトルの次元は文献 4) から 100 次元とする．100 次元で大きさが 1 のベクトルをランダムに発生させるために GNU Scientific Library(GSL)[10]の球面分布関数を用いる．球面分布関数を用いると n 次元のベクトル $(x_1, x_2, \cdots, x_n)$に対して規格化の条件

$$x_1^2 + x_2^2 + \cdots + x_n^2 = 1$$

を満たすように乱数が生成される．GSL も Fedora 用のパッケージをインストールして用いる．

単語にランダムなベクトルを割り当てただけでは，文書における単語のつながりは考慮されない．単語のつながりを取り入れるため i 番目の単語ベクトル$\mathbf{x}(i)$に対して前後に$\mathbf{a}(i)$と$\mathbf{b}(i)$を付け加える．ここで$\mathbf{a}(i)$と$\mathbf{b}(i)$は 100 次元のベクトルで$\mathbf{a}(i)$は$\mathbf{x}(i)$の直前に出てくるすべての単語ベクトルの平均である．また，$\mathbf{b}(i)$は$\mathbf{x}(i)$の直後に出てくるすべての単語ベクトルの平均である．最後に単語ベクトル$\mathbf{X}(i)$は

$$\mathbf{X}(i) = \{\mathbf{a}(i), 0.2\,\mathbf{x}(i), \mathbf{b}(i)\}$$

で表される．$\mathbf{a}(i)$は単語の前の文脈，$\mathbf{b}(i)$は単語の後ろの文脈を平均的に表わす量になっている．0.2 は前後関係を学習したベクトル$\mathbf{a}(i)$と$\mathbf{b}(i)$と$\mathbf{x}(i)$とのバランスを調整する量で WEBSOM と同じ値とする．$\mathbf{X}(i)$は 300 次元のベクトルで表される．

2.4 SOM による単語の分類

2.3 節で作成した単語ベクトルを SOM によって分類する．分類には Kohonen らによる SOM_PAK[11]を用いた．SOM は文献 7)と同じ 30×30 ユニットとし，最近接に 6 個のユニットがある六方格子とする．学習原理は次の通りである

1) 各ユニットに割り当てるベクトルをランダムに生成する．ベクトルは単語ベクトルと同じ 300 次元である．
2) 単語ベクトルによって各ユニットのベクトルを更新する．このとき，ユニットの中から最も単語ベクトルに近いユニットを選び，その近傍のユニットほど学習する単語ベクトルに近づくよう更新する．

表 1 Security-Quickstart-Redhat-HOWTO に近い文書を検索した結果（各乱数）.

	乱数 1	距離	乱数 2	距離	乱数 3	距離
1	Security-Quickstart-HOWTO	0.051684	Security-Quickstart-HOWTO	0.047248	Security-Quickstart-HOWTO	0.043415
2	Secure-Programs-HOWTO	0.580586	Secure-Programs-HOWTO	0.550175	Firewall-HOWTO	0.508161
3	Firewall-HOWTO	0.588078	Security-HOWTO	0.556794	Secure-Programs-HOWTO	0.513066
4	Wearable-HOWTO	0.597303	DSL-HOWTO	0.579209	NET3-4-HOWTO	0.528070
5	NET3-4-HOWTO	0.602009	BadRAM-HOWTO	0.582144	Security-HOWTO	0.529033
6	Security-HOWTO	0.613766	*	0.593869	IP-Masquerade-HOWTO	0.531848
7	*	0.632705	Multi-Disk-HOWTO	0.594248	Multi-Disk-HOWTO	0.539991
8	Ecology-HOWTO	0.636962	Text-Terminal-HOWTO	0.610763	**	0.540819
9	LAN-mini-HOWTO	0.638803	IP-Masquerade-HOWTO	0.619097	LAN-mini-HOWTO	0.551864
10	Adv-Routing-HOWTO	0.642380	Modem-HOWTO	0.620551	Parallel-Processing-HOWTO	0.552362
	乱数 4	距離	乱数 5	距離	乱数 6	距離
1	Security-Quickstart-HOWTO	0.044321	Security-Quickstart-HOWTO	0.056932	Security-Quickstart-HOWTO	0.046700
2	Secure-Programs-HOWTO	0.476606	NET3-4-HOWTO	0.623772	Secure-Programs-HOWTO	0.474842
3	NET3-4-HOWTO	0.494775	Secure-Programs-HOWTO	0.624335	NET3-4-HOWTO	0.512815
4	Security-HOWTO	0.510679	DSL-HOWTO	0.628045	Security-HOWTO	0.517348
5	Tips-HOWTO	0.518928	Firewall-HOWTO	0.628188	Multi-Disk-HOWTO	0.528232
6	Firewall-HOWTO	0.521880	Wearable-HOWTO	0.638703	Ecology-HOWTO	0.542292
7	Adv-Routing-HOWTO	0.522644	Security-HOWTO	0.649286	Kernel-HOWTO	0.562502
8	LFS-BOOK	0.523476	*	0.665520	Wearable-HOWTO	0.571762
9	Config-HOWTO	0.531694	Multi-Disk-HOWTO	0.666991	GCC-SIG11-FAQ	0.578976
10	Multi-Disk-HOWTO	0.533539	LAN-mini-HOWTO	0.682276	Tips-HOWTO	0.580284
	乱数 7	距離	乱数 8	距離	乱数 9	距離
1	Security-Quickstart-HOWTO	0.048423	Security-Quickstart-HOWTO	0.037442	Security-Quickstart-HOWTO	0.046501
2	NET3-4-HOWTO	0.526394	Secure-Programs-HOWTO	0.394833	Secure-Programs-HOWTO	0.542611
3	Secure-Programs-HOWTO	0.538396	IP-Masquerade-HOWTO	0.408113	Multi-Disk-HOWTO	0.585342
4	DSL-HOWTO	0.551978	NET3-4-HOWTO	0.410336	NET3-4-HOWTO	0.587074
5	Security-HOWTO	0.569434	Security-HOWTO	0.411665	Firewall-HOWTO	0.588215
6	Ecology-HOWTO	0.587570	Firewall-HOWTO	0.437258	Security-HOWTO	0.591428
7	*	0.590773	*	0.443504	IP-Masquerade-HOWTO	0.604942
8	LAN-mini-HOWTO	0.602769	LAN-mini-HOWTO	0.454058	Ecology-HOWTO	0.620167
9	Multi-Disk-HOWTO	0.603203	Ethernet-HOWTO	0.455064	DOS-Win-to-Linux-HOWTO	0.623132
10	Modem-HOWTO	0.604014	DSL-HOWTO	0.455990	Adv-Routing-HOWTO	0.633495

	乱数 10	距離
1	Security-Quickstart-HOWTO	0.045932
2	Secure-Programs-HOWTO	0.498841
3	Security-HOWTO	0.523405
4	Ecology-HOWTO	0.553223
5	NET3-4-HOWTO	0.562466
6	Firewall-HOWTO	0.565028
7	DSL-HOWTO	0.567079
8	*	0.573447
9	LAN-mini-HOWTO	0.576386
10	Wearable-HOWTO	0.592810

ファイルの拡張子txtは省略している

*は Unix-and-Internet-Fundamentals-HOWTO

**は From-PowerUp-To-Bash-Prompt-HOWTO

2.5 文書のヒストグラムの作成

各文書の単語の出現頻度を基にヒストグラムを作成する．このとき 2.4 節の SOM に基づき，同じユニット上にある単語は同じものとみなす．従って，各文書は 30×30 次元，すなわち 900 次元のベクトルで表示される．j 番目の文書のヒストグラムのベクトルを$\mathbf{h}(j)$で表す．

2.6 文書間の距離の計算

2.5 節の 900 次元のベクトルを正規化して文書を表すベクトルとする．

$$\mathbf{e}(j) = \frac{\mathbf{h}(j)}{|\mathbf{h}(j)|}$$

これを基に文書 j と文書 k の距離を表す行列D_{ij}を

$$D_{jk} = |\mathbf{e}(j) - \mathbf{e}(k)|$$

で定義する．j を決めたとき，k を 1 から 384 まで変化させ，j 以外で最も小さくなるものが近い文書である．

計算には OS が Linux のディストリビューションの 1 つである Fedora 13[12]のマシンを用いた．

3 結果と考察

文書の数は 384 個あるので，すべての結果を表示できない．一例として Security-Quickstart-Redhat-HOWTO に近い文書から 10 個表示している．乱数の初期値によって結果が異なるので 10 個の異なる初期値の結果を表 1 に示す．乱数は単語ベクトルを生成するときと，SOM による単語の分類を行うときの SOM の初期化の時に用いられる．非常に距離の近い Security-Quickstart-HOWTO と Security-Quickstart-Redhat-HOWTO はほとんど同じ文書で Redhat に関する一部の記述だけが違うのみであり，どの乱数においても距離的に非常に近くなっていることがわかる．また，内容の近い Security-HOWTO もどの結果にも含まれており，ある程度，内容の近い文書が検索できることがわかる．これら 10 回の結果で平均の順位が最も低い順にならべた結果が表 2 である

表 2 Security-Quickstart-Redhat-HOWTO に近い文書を検索した結果（総合）

1	Security-Quickstart-HOWTO
2	Secure-Programs-HOWTO
3	NET3-4-HOWTO
4	Security-HOWTO
5	Firewall-HOWTO
6	Unix-and-Internet-Fundamentals-HOWTO
7	Ecology-HOWTO
8	Multi-Disk-HOWTO
9	LAN-mini-HOWTO
10	IP-Masquerade-HOWTO

Linux 関連の文書には，専門用語も多く，専門用語も含まれる辞書を用いるのが理想的であるが，辞書を用意するのは非常に手間のかかる作業である．今回用いた IPA の辞書は，専門用語の含まれていない辞書であるが，それでも文書の検索が行える．

今回分かち書きに用いた MeCab であるが，Linux のコマンドである bzip2 などの文字列を bzip と 2 に分けてしまう欠点がある．MeCab は辞書で分かち書きをコントロールする方針で開発されており，アルファベットと数字の組み合わさった文字列を 1 つの単語として扱うことはできなかった．

4 おわりに

今回用いた JF のドキュメントは Linux の日本語の文書の集まりで，共通の目的をもって書かれた文書であった．今後，さまざまな文書が混ざった時にこの方法が有効であるかどうかを検証したい．

参考文献

1) http://www.google.co.jp/
2) http://www.namazu.org/
3) Kohonen, T., *Self-Organizing Maps,* Springer, New York, 2000.
4) Honkela, T., Kaski, S., Lagus, K., Kohonen, T.

Exploration of full-text databases with self-organizing maps., Proceedings of ICNN'96 IEEE International Conference on Artificial Neural Networks, Volume1, IEEE Service Center, Piscataway, NJ(1996), 56－61

5) Kaski, S., Honkela, T., Lagus, K., and Kohonen, T., Creating an order in digital libraries with self-oraganizing maps, Proceedings of WCNN'96, World Congress on Neural Networks, September 15-18, San Diego, California, Lawrence Erlbaum and INNS Press, Mahwah, NJ, (1996) 814-817

6) 岸田和明，文書クラスタリングの技法：文献レビュー，Library and Information Science, No 49(2003), 33-75

7) 尾崎数也，藪兼智英，井上正人，前原俊信，岡隆光，自己組織化マップを用いた日本語処理の試み，社会情報学研究（呉大学　現広島文化学園大学），Vol. 9(2003), 99－106

8) http://archive.linux.or.jp/JF/

9) http://mecab.sourceforge.net/

10) http://www.gnu.org/software/gsl/

11) http://www.cis.hut.fi/research/som_pak/

12) http://fedoraproject.org/