係り受けの確率的クラスタリングを用いた大規模類似語リストの作成

風間 淳一　Stijn De Saeger　鳥澤 健太郎　村田 真樹
{kazama, stijn, torisawa, murata}@nict.go.jp
情報通信研究機構　MASTAR プロジェクト 言語基盤グループ

1 概要

係り受けの大規模な確率的クラスタリングの結果から，高精度で大規模な名詞の類似語リストを生成する方法について述べる．本研究では，クラスタリングの結果得られるクラス所属確率の間の Jensen-Shannon ダイバージェンスを利用して語間の類似度を定義し，類似語リストを生成した．現時点で，語彙数を 100 万としたクラスタリングの実行に成功し，そこから 100 万語の各々の語に対して 500 個の類似語を類似度付きで生成することに成功した．本稿では，高精度なリストを生成するためのクラスタリングにおける工夫や大規模なリストを現実的な時間で生成するための近似手法について述べ，生成された類似語リストの精度評価について報告する．

2 はじめに

語の間の意味的な類似度を計算すること（あるいは，それと表裏一体の関係にある語のクラスタリング）は，人のもつ基礎的な能力とも考えられ学問的にも興味深く，また，クエリ拡張，スムージング，素性生成など，自然言語処理での幅広い利用が考えられることから，これをコーパスからの統計をもとに自動で計算する方法が数多く提案されてきた．

本研究の目的は，近年爆発的に増加した Web 上の文書を用いて，そのような膨大な現実世界の言語情報を処理するのに十分なカバレッジをもつ 100 万語超の大規模な語彙に対して語間の類似度を計算し，それによって類似語リストを生成することである．本稿で述べる手法を用いて生成された類似語リストの小規模なバージョン（50 万語）は，我々が開発している検索支援システム「鳥式改」[8] での「アナロジー推論」のために既に利用されており，規模がアプリケーションの質を変え得るという示唆が得られている．また，本研究で生成したクラスモデルは，[7] の研究など，我々が行う様々な研究で利用され始めている．我々は，これらのクラスモデルや類似語リストを内部的な利用にとどめるのではなく，今年設立を予定している「高度言語情報融合フォーラム」を通じて一般に配布し，様々な利用に供したいと考えている．

これまで提案されている類似度の計算方法の多くは，「意味的に似た語は似た文脈で出現する」という分布仮説に基づいている．特に，周辺に出現する語や係り受け関係にある語で文脈を定義し，その語との共起頻度などをベクトルで表してその間の類似度を何らかの方法で計算する手法が多い [1, 10, 9]．本研究で用いる方法も，基本的にはこれらの手法と共通するが，共起頻度からベクトルを直接作るのではなく，確率的クラスタリングを間に挟むところが多少異なる．まず，名詞と動詞の係り受けを隠れクラスをもつ確率モデルで表現し，それを EM アルゴリズム用いて推定するクラスタリング [6] を実行する．その結果，名詞に対してその隠れクラスへの所属確率の分布 ($p(c|n)$) を得，その分布の間の Jensen-Shannon ダイバージェンスを計算し，そこから類似度を得る．隠れクラスの数は通常語彙数よりも小さい（数千程度）ため，隠れクラスを用いて文脈をスムージングしていることになる．スムージングなどによるデータスパースネスの回避は質の高い自然言語処理を行うのに重要であることが知られており，本研究でもその知見をこのような形で取り入れることにした．また，ただ語間の類似度を計算するよりも，クラスタリングを行うことによって係り受けの確率モデルが得られ，たとえば係り受けの語義曖昧性解消など，様々な応用での利用もできるという利点もある．

どのような手法を用いるにしても，100 万語規模の類似語リストの作成は，計算量的にも，また，後で述べるように精度面でも，それほど簡単ではない．近年，大規模な類似度計算を指向した研究はいくつかあるが [3, 9]，我々の知る限り，我々と同等の規模での計算を実際に行ったとする例は見つからない．我々は，まず，[2] において，[6] のクラスタリングアルゴリズムを MPI を用いて分散・並列化し，50 万語，3,000 クラスという大規模なクラスタリングが現実的なメモリ量と時間（8GB ノード × 8 の環境で約 1 週間）で行えることを示し，クラスモデルの利用例として，固有表現認識での辞書素性として利用することで精度を向上できることを示した．また，その後，プログラムの頑健性を改良することでより大規模なクラスタリングも可能にした[*1]．その結果，現時点で 100 万語，2,000 クラスでのクラスタリングに成功しており，本研究ではそれを用いる．

ここから，上で述べたように類似度リストを作成するのであるが，研究の初期の時点で，そのままでは結果の類似度リストに多くのノイズ的な語が含まれてしまうことが分かった．データの観察から，Web データ特有の現象としていくつかの（通常低頻度と考えられる）語の係り受けの頻度が極端に大きくなるということがあり，それがクラスタリングのノイズになっていることが推測された．そこで，多少アドホックではあるが，係り受けの頻度をそのまま用いるのではなく対数を取った上でクラスタリングを行うという手法を試みた．その結果，類似語リストの精度が明らかに改善されることが分かった．

また，類似語リストの生成は単純には全ての語の組み合わせについて Jensen-Shannon ダイバージェンスを計算すればよいのであるが，それも計算量が非常に大きい．本研究では，距離を計算する対象をある程度ヒューリスティクスで限定して近似的に計算し，また，計算自体を並列化することによって現実的な時間で計算できるようにした．さらに，これらの手法を基にさらに精度の高いリストを生成するために，複数のクラスタリング結果を組み合わせるという手法を開発した．

3 手法

3.1 係り受け関係のクラスタリング

本研究で用いるクラスタリング [6] では，以下のような隠れクラスモデルを仮定する．

$$p(n, v, r) = \sum_c p(n|c)p(\langle v, r\rangle|c)p(c). \quad (1)$$

[*1] 具体的には，計算途中のパラメータを一定反復ごとに保存するような機能などを実装した．不意のノード故障や他ユーザの思わぬ割り込みによって全ての結果が無駄にならないようにするためである．

ここで，$v \in \mathcal{V}$ は動詞，$n \in \mathcal{N}$ は v と係り受け関係 r にある名詞，$c \in \mathcal{C}$ は隠れクラスである．名詞 n は，複合語や修飾語の付いた名詞を含む．関係 r は，名詞につづく助詞で表す．「れる」などの助動詞は名詞 n を大きく変化させるので，動詞 v の一部としてつなげて用いる．また，「N_1 の N_2」という名詞-名詞の係り受けも，$v = N_2, r = \text{の}, n = N_1$ と考えることで，クラスタリングに用いる．これは，このような表現中の N_2 によっても，N_1 の性質がよく表わされるからである．

クラスタリングの学習データは $\{(n_i, vt_i, f_i)\}_{i=1}^{L}$ として与えられる．$vt \equiv \langle v, r \rangle \in \mathcal{VT}$ であり，これを動詞テンプレートと呼ぶ．f_i は，係り受け (n_i, vt_i) がコーパスに現れた回数である．EM アルゴリズムによるクラスタリングでは，この学習データの対数尤度を最大にするような $p(vt|c), p(n|c), p(c)$ を反復により求める [6]．このクラスタリングアルゴリズムを，[2] で述べられているように分散・並列化して用いる．

本研究では，学習データとして，検索エンジン研究基盤 TSUBAKI [5] の約 1 億ページ・60 億文のデータから係り受け関係を抽出した．約 21 億種類の係り受けが頻度とともに抽出され（内，15.8 億が名詞-動詞，5.3 億が名詞-名詞の係り受け），そこには，頻度 1 のものを除いた状態で，約 8,400 万種類の名詞があり，約 2,500 万種類の動詞テンプレートが含まれていた．この抽出自体も計算量が大きく，それほど容易ではない．我々は，InTrigger プラットホーム[*2]などのクラスタ環境を活用してこれらの処理を行っている [4]．

このデータから，さらに名詞および動詞テンプレートを 100 万語に絞り，係り受け関係を抽出した．これには [2] の時と同じように，ある語が出現する係り受けペアの種類数によって語をソートして上位 100 万語を選択することによって行った[*3]．最終的には，選択した語のみを含む係り受けペアを抜き出して，約 4.3 億ペア，ファイルの容量にして 13 GB の学習データを得た．

3.2 係り受け頻度の補正

はじめに述べた通り，コーパスから得られた係り受けの頻度をそのまま用いた場合，Web データ特有のノイズのため極端に頻度が大きくなった語のためにクラスタリングの結果が乱されることが観察された．確定的な理由は不明であるが，クローリングの際の重複獲得，あるいは大量に自動生成されるようなページ固有の表現が原因ではないかと考えられる．

本研究では，この問題に対処するため，コーパスから得られた頻度 f_i の代わりに，$\log(f_i) + 1$ を学習の際に用いることを試みた．実験で示すように，この方法により類似語リストの精度は大幅に改善される．頻度の対数をとることは，情報検索の用語の重み付けでしばしば用いられており，共起ベクトルに基づいて類似度を計算する際に利用することで精度が向上したという報告もある [10]．ただし，これらは我々のように Web データに起因するノイズを取り除くということを動機としたものではない．また，本来は正しい頻度に基づいて最尤推定を行うことを前提とするアルゴリズムにおいてこのように補正をかけた頻度がうまく働くということは，他の最尤推定に基づく様々な学習への示唆を含んでいるものと考えている．

また，前節最後で述べた 100 万語の選び方は，一見，単純に頻度でソートする方法よりも奇妙に思われるかもしれないが，この方法を取らないと結果はさらに悪化することが分かっている．処理上簡単であったためそのような方法を取っていたのであるが，結果的には，ここで述べた頻度の補正と同じようなノイズの除去の効果があると考えられ，また，将来的には，これらの知見を総合して直接 $log(f_i)$ でソートして上位の語を選択するという方法をとることも考えられる．

3.3 類似語リストの生成

本研究では，語の間の類似度の計算には，クラスタリングの結果から計算される語のクラス所属確率分布 $p_i(c) \equiv p(c|n_i) = p(n_i|c)p(c)/p(n_i)$ の間の距離を計算して利用する．確率分布間の距離としては KL ダイバージェンス $D_{KL}(p_1||p_2) = \sum_{i=1}^{C} p_1(c_i) \log \frac{p_1(c_i)}{p_2(c_i)}$ がよく知られているが，これには，どれか一つの c_i でも $p_1(c_i) \neq 0$ かつ $p_2(c_i) = 0$ となると全体の距離が無限大になってしまい距離の比較が上手くできないという問題がある．後で述べるように，クラスタリング結果のパラメータは疎になっており，この問題は実際に起こる．そこで，本研究ではこの問題のない以下のような Jensen-Shannon ダイバージェンスを用いる．

$$D_{JS}(p_1||p_2) = \frac{1}{2}(D_{KL}(p_1||\frac{p_1+p_2}{2}) + D_{KL}(p_2||\frac{p_1+p_2}{2}))$$

$\frac{p_1+p_2}{2}$ は p_1，p_2 をベクトルとしてみたときの平均である．Jensen-Shannon ダイバージェンスは，[1] などで共起に基づいた語の類似度計算に用いられ，その有効性が示されている．本研究では，これをクラスタリングで得られるクラス所属確率分布に適用した．距離が計算できれば，n_1 と n_2 の類似度は $sim(n_1, n_2) = -D_{JS}(p_1||p_2)$ のようにすればよい．

名詞 n_1 の類似語リストは，他の n_2 との類似度を計算し，その上位 K 語を取り出せば生成できる．しかし，全ての n_2 との類似度を計算するのは現実的ではない．幸いなことに，学習結果の $p(n|c)$ は値が極小さい要素（本研究では 10^{-6} より小さい要素）を捨てればかなり疎になる（ゼロでない要素が全体の 3%-5%）．その結果 $p(c|n)$ も疎になる．本研究では，この性質を利用して以下のような近似計算を行うことにした（M は近似のパラメータ）．

1. $p(c|n_1)$ がゼロでないような c を最大 M 個取り出す．
2. 上の c 各々について，$p(c|n_2)$ がゼロでない n_2 を $p(c|n_2)$ が大きいものから最大 M 個取り出して候補に加える（重複は除く）．– (A)
3. 候補の n_2 に関してだけ $sim(n_1, n_2)$ を計算し，ソートして最大 K 個取り出す．

これにより，計算量を大幅に減らすことができる．この方法を「**手法 A**」と呼ぶ．さらにこれの改良として，上の (A) の部分を「$p(c|n_2)$ がゼロでない n_2 を $p(c|n_2)$ が $p(c|n_1)$ に近いものからその周囲の最大 M 個取り出す」と変えたものも試した．これを「**手法 B**」と呼ぶ．これは，$p_1(c_i)$ と $p_2(c_i)$ が近いほど距離への影響が大きいからである．$p(c|n_2)$ はあらかじめソートしておくことができ，値が近いものを探すのは二分探索などにより比較的低コストで行うことができる．また，さらに精度を向上させる試みとして，J 個のクラスタリング結果があったときに，各モデルで手法 A と同様に候補を取り出して，最後に $sim(n_1, n_2) = \sum_j sim_j(n_1, n_2)/J$ で類似度を計算するという「**手法 C**」も試した．$sim_j(n_1, n_2)$ は j 番目のモデルで計算した類似度である．これは，EM アルゴリズムによるクラスタリングは与えられた初期値[*4]によって結果が変わる局所最適なアルゴリズムであり，複数の初期値によって作られた複数のモデルの情報を利用することで，より精度が向

[*2] 科学技術研究費特定領域 「情報爆発時代に向けた新しい IT 基盤技術の研究」において構築されているクラスタ環境である

[*3] [2] では記述もれのため明示されていなかった．

[*4] 現在は Dirichlet 事前分布 ($\boldsymbol{\alpha} = \mathbf{1.0}$) からランダムに与えている．

リスト名	モデル，手法	生成時間	近似率
L1	**lf1**, **手法 A** (M=400)	55.5 h	0.0323
L2	**lf2**, **手法 A** (M=400)	-	-
L3	**lf2**, **手法 A** (M=800)	107.7 h	0.0621
L4	**nlf**, **手法 A** (M=400)	-	-
L5	**lf2**, **手法 B** (M=400)	62.3 h	0.0305
L6	**lf1**,**lf2**, **手法 C** (M=400)	181.2 h	0.0269
L7	**lf1'**, **手法 A** (M=400)	-	-

表 1　生成した類似語リスト

リスト名	MAP	MTP5	MTP10	MTP20
L1	0.179	0.629	0.618	0.603
L2	0.255	0.660	0.651	0.639
L3	0.273	0.671	0.659	0.645
L4	0.123	0.550	0.535	0.518
L5	0.259	**0.685**	**0.672**	**0.658**
L6	**0.274**	0.666	0.661	0.651
L7	0.170	0.666	0.650	0.626

表 2　[11] のデータでの評価結果

上すると考えたからである．最後に，各名詞に対するリスト作成は独立であるので，並列化することができる．本研究では OpenMP を利用して複数スレッドでの実行を実装した．

4　実験

まず，100 万語 ×2,000 クラスのクラスタリングを何通りか行い，以下のモデルを作成した．

- 頻度補正あり，初期値 A で学習，反復数 150 – **lf1**
- 頻度補正あり，初期値 B で学習，反復数 150 – **lf2**
- 頻度補正なし，初期値 B で学習，反復数 150 – **nlf**
- 頻度補正あり，初期値 B で学習，反復数 300 – **lf1'**

初期値 A と初期値 B では確率モデルの初期パラメータを生成する乱数生成器に与えるシードが異なっている．それぞれ，InTrigger 環境（Intel Xeon Quad 2.33Ghz 環境) で 16 並列での実行を行い，約 1 週間で結果を得た．(ただし，**lf1'** は約 2 週間かかる）そして，これらのモデルと類似語リストの生成方法を様々に組み合わせて表 1 にあるリストを生成した．全て出力数 K は 500 である．表中，生成時間は，Intel Xeon Quad 2.33Ghz 環境で 8 並列で計算した場合である．近似率は，単純に全てのペアを計算した場合に比べてどのくらいの計算量で収まっているかを示す値である．ここから，L1 の生成には，並列化がなければ約 573 日，さらに近似がなければ約 17,740 日かかることが分かり，いかに計算量が大きいかが分かる．また，手法 A は，M にほぼ比例して計算量が大きくなること，手法 B で加えられた処理のコストはそれほど大きくないこと，手法 C は二つのモデルを用いる分だけ計算量も増えていることが分かる．

4.1　**実験** 1

まず，自動で行える評価として，[11] の用語抽出用評価データを用いた評価を行った．このデータの内「一項データ」と呼ばれるデータは，「国名」など，（ある時点で）比較的簡単に列挙しつくせる固有表現の集合 58 種類を人手で作成したものである．このような集合中の語の類似語を求めれば，集合中の他の語が類似語として上位に得られるのがひとつの理想的な状態であろう．今回得られた類似語リストがそのようになっているかを評価した．このデータの内完全に列挙し尽くされているとフラグの付いている 45 集合の中の語の内，複数の集合に入っていない曖昧性のない語，6,829 語についてそれを調べた結果が表 2 である[*5]．表中 MAP は，[11] と同じように average precision の平均，MTP5, 10, 20 とあるのは，第 5, 10, 20 位までの精度の平均である．新聞のデータを用いた [11] に比べて全般的に良い性能が出ている（[11] で報告されている MAP の最大値は 0.206，MTP5 の最大値は 0.490）．実験条件が完全に同じではないため，単純な比較はできないが，[11] では入力に 5 個のシードを用いていたことを考慮すると，今回の類似語リストは精度が良いと言える．総合的に見ると，**L5**（**手法 B**）, **L6**（**手法 C**）とも精度向上の効果があり，また，**L2** と **L3** を比べれば，M を大きくすることによって計算量は大きくなるが精度も良くなるという傾向が分かる．**手法 B** と**手法 C** の優劣はこの段階では付け難いが，計算量を考えると**手法 B** のほうが良いと言えるかもしれない．また，**L5**（**手法 B**，M=400）と **L3**（**手法 A**，M=800）の優劣も付け難いが，計算量を考えると**手法 B** (M=400) のほうが良いと言えるかもしれない．また，**L4** の精度は低く，対数による頻度の補正の効果はかなり大きいことが分かる．**L1** と **L2** の結果を見ると，初期値によって精度はかなり変わりそうである．また，クラスタリングの反復回数は **lf1** などの 150 回で十分のようである．

4.2　**実験** 2

実験 1 では，要素が列挙できるようなタイプの固有表現集合しか評価できないのに加え，類似度は実験 1 のように対象語と類似語が同じ上位語を持つような場合にのみ高いのが理想という訳でもない．そこで，100 万語から 200 語を対象語としてランダムに選び，これらの語の類似語リストに対して，著者の一人（第一著者）が人手による主観評価を行った．各手法によるリストの上位 20 語をマージして重複を除いた上で順番をランダム化し，どの手法の出力か分からないようにして，以下のような規準で類似語の評価を行った．

5 点 (same)　異表記や省略などの同義語と考えられるもの．
4 点 (very similar)　良く似ているもの．
3 点 (similar)　似ているもの．
2 点 (may similar)　似ているとも言えるもの．
1 点 (not similar)　似ていないもの．
0 点 (bad)　形態素解析誤りなどの単語として不適切なもの．

また，200 語自体も bad かどうか判定して，bad になった 17 語は評価から除いた（100 万語の内，約 8.5% は不適切な語が含まれていることになる）．単語の意味が分からないときは，Web 検索や Wikipedia 等を利用して意味を調べた．表 3 は，3 点以上を正解と考えたときの MTP5, 10, 15, 20 をまとめたものである．また，表 4 は，表 3 での MTP5 の優劣が統計的にどのくらい有意かを見るため，ブートストラップ法（反復数 10,000）により勝率を計算したものである（縦のリストが横のリストに勝つ確率）．この結果から，複数モデルを組み合わせる **L6**（**手法 C**）が統計的にもほぼ確実に最も精度が良く，また **L4** の結果から，頻度の補正がない場合はほぼ確実に精度が最も悪いことが分かる．**L5**（**手法 B**）と **L2**（**手法 A**）との比較から**手法 B** はそれなりの確率で効果があり（ただし，MTP20 のところでは差は殆どない），また，**L5**（**手法 B**，M=400）と **L3**（**手法 A**，M=800）の優劣は付け難いが，わずかに **L3** のほうが優勢のように見える．**L1** と **L2** の結果を見ると，初期値によって精度はやはり変わりそうで

[*5] ある表現の異表記も入っている場合があるが，その情報は今回は用いなかった．

リスト名	MTP5	MTP10	MTP15	MTP20
L1	0.817	0.794	0.773	0.764
L2	0.827	0.809	0.794	0.788
L3	0.839	0.816	0.812	0.802
L4	0.658	0.628	0.605	0.593
L5	0.836	0.813	0.797	0.782
L6	**0.870**	**0.845**	**0.833**	**0.821**
L7	0.816	0.798	0.783	0.770

表 3　人手による主観評価の結果

	L1	**L2**	**L3**	**L4**	**L5**	**L6**	**L7**
L1	-	0.299	0.126	1.0	0.164	0.0001	0.511
L2	0.681	-	0.07	1.0	0.190	0.0017	0.713
L3	0.862	0.910	-	1.0	0.623	0.0047	0.899
L4	0.0	0.0	0.0	-	0.0	0.0	0.0
L5	0.824	0.779	0.333	1.0	-	0.0034	0.845
L6	0.9998	0.998	0.994	1.0	0.995	-	0.9996
L7	0.464	0.269	0.092	1.0	0.143	0.0003	0.0

表 4　MTP5 の統計的有意性のテスト

あり，統計的に確実でないが **L2** のほうがわずかに精度が良さそうである．また，**L7**（反復数 300) の結果を見ると，クラスタリングの反復回数は 150 回で十分のようである．複数モデルを組み合わせる **L6**（**手法 C**）の精度の良さの理由を考えたときに，実質的な隠れクラスの数が 2 倍になっているからではないかという疑問が起きる．これに答えるには，4,000 クラスでのクラスタリングを行えばよいのであるが，計算資源の問題からまだ実行するには至っていない．3,000 クラスの結果は，実験 2 終了後に得られたため，ここでの比較はできないが，実験 1 での結果は，MAP=0.206, MTP5=0.674, MTP10=0.663, MTP20=0.650 となった．これを表 2 の **L6** と比較すると，MAP はかなり悪く，MTP5,10 がわずかに良く，MTP20 はほどんど差はないという結果になった．MAP に関しては単純にクラス数を増やしても改善できなそうであるが，これだけから結論を出すことはできないため，さらに実験を進めていきたい．ただし，**手法 C** では，クラス数の小さいモデルを複数回に分けて作成するので，よりメモリ資源の小さい環境でも高精度なリストが作成できるという利点はありそうである．また，今回有効であることが分かった**手法 B** と**手法 C** の組み合わせなども今後試していきたいと考えている．また，現在用いられていない「A などの B」や「A への B」といった多様な係り受けを抽出して利用することでより細かい意味の判別ができるようにしたい．頻度の補正については，Okapi (BM25) の $f/(1+f)$ とった形式の他の補正式なども考えられる．また，対数で補正したクラスタリング結果を [2] の固有表現認識で用いても，精度は向上せず，むしろ極わずかだが低下するという現象も観察されている．これらのことから，頻度の補正については更なる検証が必要であると考えている．

最後に，表 5 に **L6**（**手法 C**）での類似語の例を示す．

5　今後の課題とまとめ

本研究では，係り受けの大規模なクラスタリング結果を用いて 100 万語という大規模な語彙に対して高精度の類似語リストを効率的に生成する方法を提案した．我々の知る限りこのような大規模な類似語リストを生成した研究は他にない．

実験では，提案した頻度の補正や複数モデルの組み合わせによって精度が向上することを示した．しかし，評価結果は予備的な段階にあると考えており，より良い評価方法も模索していきたい．まず，実験 2 の結果は著者一人の評価であり，

対象語 (id)	類似語
絨毯 (10000)	じゅうたん, カーペット, ジュータン, 畳, クッション, 敷物, タタミ, たたみ, ジュウタン, マット, テーブルクロス, フローリング, カーテン, 人工芝, 畳み, 布地, 座布団, ...
ＳＣＥ (20000)	任天堂, セガ, コナミ, ＳＣＥＩ, スクウェア, スクエニ, カプコン, ナムコ, ＳＥＧＡ, バンダイ, コーエー, ハドソン, エニックス, スクウェアエニックス, タイトー, テクモ, ＥＡ, ...
メモリボード (500083)	ＤＩＭＭ, ＰＣＩボード, 拡張カード, 拡張ボード, ＳＩＭＭ, ＣＰＵカード, ＰＣＩカード, ロジックボード, ＬＡＮボード, ＶＧＡカード, ＳＣＳＩボード, 増設カード, ...
一般検診 (700004)	基本検診, 人間ドッグ, 一般健診, 妊婦検診, 基本健診, 脳ドック, 市民健診, 妊婦健診, 人間ドック, 一般外来, 人間ドック等, 基本健康診査, 外来診察, 3 歳児健診, 日帰り人間ドック,

表 5　類似語リストの例

その主観に大きく影響されているため，複数人での評価を行いたいと考えている．また，類似性の判断には多種多様な側面があり，今回の人手評価を行った第一著者が振り返ると，

- 対象語と類似語が同じ上位語を持つか
- 複数の上位語があるときにいくつ共通するか
- 上位下位関係にあるか
- 反対の意味か（この場合類似度が高いと判定した）
- 同じ分野の関連語か
- 表層的に似ているか

のような様々な要因を総合して判断していたように思う（明確なルールを設けていた訳ではない）．また，same と bad を除けば 4 段階の評価になっており，さらに細かい段階があれば差をつけられ，手法の差をより明らかにできるのではないかという例もあれば，点数では優劣を付け難い例もあった．全般的には，評価は相対的であり，評価対象になっている類似語全体をみて 4 点から 1 点の間で割り振るという傾向があったように思う．以上のことを念頭に置きながら，評価実験方法の設計や生成手法の改善などを行っていきたいと考えている．

参考文献

[1] Ido Dagan, Lillian Lee, and Fernando Pereira. Similarity-based models of cooccurrence probabilities. *Machine Learning*, 1999.

[2] Jun'ichi Kazama and Kentaro Torisawa. Inducing gazetteers for named entity recognition by large-scale clustering of dependency relations. In *ACL-08: HLT*, 2008.

[3] Deepak Ravichandran, Patrick Pantel, and Eduard Hovy. Randomized algorithms and nlp: Using locality sensitive hash function for high speed noun clustering. In *ACL 2005*, 2005.

[4] Stijn De Saeger, 鳥澤健太郎, 風間淳一. Mining web-scale treebanks. 言語処理学会第 15 回年次大会発表論文集, 2009.

[5] Keiji Shinzato, Tomohide Shibata, Daisuke Kawahara, Chikara Hashimoto, and Sadao Kurohashi. Tsubaki: An open search engine infrastructure for developing new information access. In *IJCNLP 2008*, 2008.

[6] K. Torisawa. An unsupervised method for canonicalization of Japanese postpositions. In *NLPRS 2001*, 2001.

[7] 山田一郎, 鳥澤健太郎, 風間淳一, 黒田航, 村田真樹, Francis Bond, 隅田飛鳥. 統語構造の特徴を利用した上位下位関係辞書の拡張. 言語処理学会第 15 回年次大会発表論文集, 2009.

[8] 鳥澤健太郎, 隅田飛鳥, 野口大輔, 柿澤康範, 風間淳一, Stijn De Saeger, 村田真樹, 黒田航, 山田一郎, 塚脇幸代, 太田公子. ウェブ検索ディレクトリの自動構築とその改良 – 鳥式改 –. 言語処理学会第 15 回年次大会発表論文集, 2009.

[9] 相澤彰子. 大規模テキストコーパスを用いた語の類似度計算に関する考察, 2008.

[10] 寺田昭, 吉田稔, 中川裕志. 文脈情報による同義語辞書作成支援ツール. 情報処理学会　研究報告 2006-NL-176, 2006.

[11] 村田真樹, 馬青, 白土保, 井佐原均. 用語抽出用評価データの作成とその利用. 言語処理学会第 10 回年次大会併設ワークショップ, 2004.