Webにおける名寄せシステム

小野 真吾　　吉田 稔　　中川 裕志
東京大学大学院 情報理工学系研究科　東京大学 情報基盤センター　東京大学 情報基盤センター
ono@r.dl.itc.u-tokyo.ac.jp　mino@r.dl.itc.u-tokyo.ac.jp　nakagawa@dl.itc.u-tokyo.ac.jp

1　はじめに

名寄せとは，複数存在する同一人物をまとめることである[1]．我々は，Web 上での複数の Web ページに渡って人名，地名などについての同名が出現した際，同一の実体ごとに Web ページを分類する名寄せシステムを開発している．本発表では，このシステムの内部の文書分類のアルゴリズムについて，いくつかの新しい手法を提案する．性能評価の結果，提案した手法を組み合わせた方法において，シンプルな文書分類の方法と比べて，F-measure で 0.2 以上の性能の向上が見られた

2　関連する先行研究

人名に関する名寄せを行うこと，すなわち同姓同名の同定についての試みは徐々に行われるようになっている．手法は主に 2 種類に大別される．1 つは文書中の個人情報を抽出し，それらを判定に用いる方法である．もう 1 つは Web の構造解析を応用する方法である．

個人情報を用いる方法として，[1] がある．これは，自前の情報抽出ツールを用いて文書中から個人情報を集め，ベクトル空間モデルによる類似度や固有表現の類似度とともに文書の類似度を計算して，同名を含む文書が同じ実体を示すかどうかを判定する．ただし，この方法では情報抽出ツールが必要になるほか，どれだけ個人情報が性能の向上に寄与しているかは示されていない．

Web における特徴的な構造であるリンク情報を用いるものとして，[2] が挙げられる．Web のリンク構造の解析と文書中の単語を用いたベクトル空間モデルを併用し，2 種類のクラスタリングを行うことで同姓同名の人物の分離を行う手法である．しかし，この方法においては，同じコミュニティに属す人物を同時に分離することしかできず，対象の人物が所属するコミュニティに関する前提知識を用いた名寄せの方法となっている．

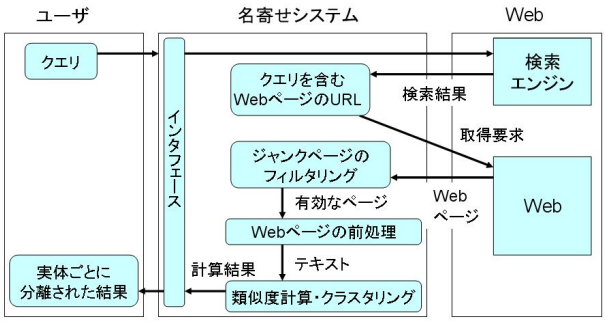

図 1. システムの概要

3　名寄せシステム

3.1　提案するシステムの概要

名寄せシステムの概要を図 1 に示す．名寄せシステムは，以下の要件を満たすシステムとして開発されている．

- Web ページを同一の実体を指すページごとに分類する[2]
- 分類において，前提知識を用いない
- 対象とするのは人名に限らず地名，組織名などの一般的な名詞も含む
- 実用に耐えうる応答時間にする

名寄せシステムは次のステップにより名寄せを行う．

1. 調べたいクエリを名寄せシステムに入力する
2. そのクエリを検索エンジンに与え，検索結果上位 k 件を得る

[1]本来，名寄せとは金融機関などにおいて同一名義の勘定をまとめることである．この名寄せが転じて上の意味となっている．

[2]なお，1 つの Web ページは 1 つの実体のみを参照していると仮定する．

3. 検索結果のページをすべてダウンロードし，形態素解析や NE のタグ付けなどをしておく[3]

4. 類似度計算およびクラスタリングを行う

5. 結果を表示する

我々はまず，ベクトル空間モデルで類似度を計算し，階層併合的クラスタリングを行う方法を実装した．しかしながら，この方法では，観測される単語の出現頻度にのみ着目し，文書に出現する単語の出現位置を考慮せずに扱うことから本来テキストが持つ多くの有用な言語情報が扱えないという問題があり，十分な精度が得られなかった．そこで，我々は対象となる文字列の近隣に出現する文字列だけを用いて類似度を計算する方法や，文書中の意味情報を持っているであろう固有表現（NE; Named Entity）を用いる類似度計算法を提案する．

また，Web ページにおいては，それぞれのページが様々な目的で作られているため，情報源として利用価値が低いページや，情報を持たないページが存在する．実際の Web ページを対象にシステムを作動させたとき，これらがノイズ的ページとして，システムの性能の低下を招くことが頻繁に起こった．そこで，名寄せシステムの性能を向上させるために，なんらかの基準でこのようなページを排除する仕組みが必要となる．

以下では，ノイズ的なページを除去した方法や文書間の類似度の新しい計算法を提案する．

3.2 ジャンクページのフィルタリング

Web 上には様々なページがあり，各ページも日々更新され続けている．そのため，意味を持たないページも多々存在し，名寄せシステムのようなシステムを作用させる上で，悪影響を及ぼす．そこで，情報を持たないページや必要ないページを省くことを考える．具体的には，以下の性質を持つページをジャンクページと定義する．

J1. リンク切れなどで見ることのできないページ

J2. 名前や数値の羅列のみで，他の情報がないページ

J3. クエリの文字列を含まないページ

J4. 複数のページを統合して表示するページ

J5. 同姓同名の複数の実体を参照するページ

J1，J2 については実際に意味を持たないページである．また，J3，J4，J5 は名寄せを行う場合に悪影響を及ぼす可能性のあるページである．上記のジャンクページを除去するために，以下のフィルタリングルールを考え，適用した．

[3]なお，形態素解析器として Sen を用いた．NE のタグ付けについても Sen の与える結果を用いている．

表 1. フィルタリングルールの性能

ルール	F-measure	Precision	Recall
F1 , F2	0.7143	0.5556 (20/36)	1.0000 (20/20)
F4	0.5379	0.3679 (78/212)	1.0000 (78/78)

F1. URL に日本語（と思われる文字列）を含むページ

F2. タイトルに “検索結果” が含まれるページ

F3. クエリの文字列を持たないページ

F4. 固有表現が非常に高頻度で出現するページ

ルール F1 と F2 はジャンクページの性質 J4 および J5 を除去することを狙いとしている．また，F3 については J1 および J3 に，F4 については J2 に対応している．

フィルタリングルールの性能を表 1 に示す．なお，各指標の計算方法は次のとおりである．ルールによってフィルタリングしたページの集合を F，フィルタリングされるべきジャンクページの集合を J とすると，Precision (P)，Recall (R)，F-measure (F) はそれぞれ

$$\mathrm{P} = \frac{|F \cap J|}{|F|},\ \mathrm{R} = \frac{|F \cap J|}{|J|},\ \mathrm{F} = \frac{2\mathrm{PR}}{\mathrm{P+R}}$$

となる．F1 と F2 については同じようなページを除去するのが目的であるからまとめて扱う．また，F3 に関しては明らかに名寄せには利用不可能なページであるため，取り除いてある．

3.3 近隣文字列のマッチング

先述のとおり，ベクトル空間モデルを用いる際には，単語の位置情報が失われてしまう．そこで，その欠点を補うべく，単語の位置情報を強く利用する方法を提案する．

procedure 近隣文字列マッチング

1. クエリ q を含むすべての文書 d_j について，以下を行う．

  1.1. 文書 d_j $(1 \leq j \leq k)$ におけるクエリ q の出現位置 p_i $(1 \leq i \leq n_j)$ をすべて検索する．

1.2. すべての p_i について，位置 $p_i-\alpha$ から $p_i+\alpha$ にある単語の集合 S_j を求める．

1.3. S_j に属す単語 a に以下の重み w_{aj} をつける．（禁止リストについては後述する）

$$w_{aj} = \begin{cases} 0 & a \text{ が禁止リストに含まれる} \\ 1 & \text{それ以外のとき} \end{cases}$$

2. すべての文書のペア d_x, d_y $(1 \leq x < y \leq k)$ について，以下の方法により，d_x, d_y においてクエリ q が示す実体が等しいかどうかの判定を行う．

2.1. 近隣文字列の類似度 $\mathrm{sim}_{\mathrm{NSM}}(d_x, d_y)$ を次のように計算する

$$\mathrm{sim}_{\mathrm{NSM}}(d_x, d_y) = \sum_{a \in S_x \cap S_y} w_{ax} w_{ay}$$

2.2. $\mathrm{sim}_{\mathrm{NSM}}(d_x, d_y) \geq \theta_{\mathrm{NSM}}$ であるならば，2 つの文書 d_x, d_y に出現するクエリ q は同一の実体を参照する．

上記のアルゴリズム中において，α はマッチングに用いる単語数，θ_{NSM} は閾値であり，これらはあらかじめ設定する．この方法では，規則をもとにしたマッチング（この場合は近隣文字列の一致）により，文書がクラスタリングされると見なすことができる．そこで，この方法を**近隣文字列マッチング**（NSM; Neighbor String Matching）と呼ぶ．

人名を例に考えると，文脈中に人名が出現した場合，肩書き，年齢などといったその人に関する情報は，人名の出現位置の近くに出現すると考えるのが自然である．すなわち，近隣文字列マッチングでは人名の出現位置のごく近傍に出現する語に着目することで，特殊な情報抽出の方法を用いることなしに，名寄せを行う上で重要な素性を抽出し，比較が行われることが期待できる．

なお，名前の前後によく現れる “氏” や “さん”，“氏名” などといった単語については禁止リストを用いて，重みを与えないようにする．

3.4 NE を用いたマッチング

Named Entity（**固有表現**，以下 NE）とは人名や組織名など事物に特有の表現である．NE は一般的な語と比較して，より文書を特徴付けていると考えられる．そこで，NE を用いて同一人物の同定をすることを考える．例えば，人名の場合，

- 名寄せ対象とは別の特定の人物と，複数の文書に渡って共に現れる（このことを，共起する，と言う）場合，それらはすべて同一人物である
- 人物の名前の近くに出現する組織名は，その人物の所属を表すと考えられ，複数の文書でそのような状況ならば，その人物は同一人物である
- 新聞記事などのように日付が付記される文書においては，同一の日の記事に出現する同名の人物は同一人物である

といったヒューリスティクスに基づく方法が考えられる．また，地名においても共起する地名や人名同様日付の情報を用いることで同一の場所の判定が行えると考えられる．

これらはすべて，必ずしも同一の実体を参照すると断定はできないが，同一の実体を参照している確率が非常に高い（すなわち，それぞれに文書が異なる実体を参照する場合が稀である）と言える．このような判断基準に基づいて，複数の文書に渡って出現する同名の参照先が同一実体であるかどうかを判定する方法を，NE によるマッチングと呼ぶことにする．NE は文書を非常に強く特徴付けたり，文書中のメインとなるトピックを与えるものであるため，NE を正しく用いることができれば，非常に強力な同一の実体の判定方法となることが期待できる．

名寄せシステムで用いている NE のマッチングのアルゴリズムを以下に示す．

procedure NE マッチング

1. クエリ q を含むすべての文書 d_j $(1 \leq j \leq k)$ について，NE tagger によってタグ付けされた人名，および地名を抽出する．なお，人名に関しては姓名が揃って出現するものに限る．

2. NE の類似度 $\mathrm{sim}_{\mathrm{NE}}(d_x, d_y)$ を次のように計算する．

$$\begin{aligned} &\mathrm{sim}_{\mathrm{NE}}(d_x, d_y) \\ &= \beta * (d_x, d_y \text{に共通して出現した人名の数}) \\ &+ \gamma * (d_x, d_y \text{に共通して出現した地名の数}) \end{aligned}$$

3. $\mathrm{sim}_{\mathrm{NE}}(d_x, d_y) \geq \theta_{\mathrm{NE}}$ であるならば，2 つの文書 d_x, d_y に出現するクエリ q は同一の実体を参照する．

なお，θ_{NE} はあらかじめ定めた閾値，β,γ は重み付けのパラメータである．

4 性能評価

複数の文書についてクラスタリングを行った場合の評価は以下のような方法で行う [3].

正解集合を $\mathcal{C} = \{C_1, C_2, \cdots, C_n\}$，クラスタリングを行った結果の集合を $\mathcal{D} = \{D_1, D_2, \cdots, D_m\}$ とする．ここで，C_i や D_j は文書の集合（クラスタ）である．正解集合の各クラスタ $C_i(1 \leq i \leq n)$ について，Precision，Recall，F-measure の値 P_{ij}，R_{ij}，F_{ij} をすべての結果のクラスタ $D_j(1 \leq j \leq m)$ に対して計算する．

$$\mathrm{P}_{ij} = \frac{|C_i \cap D_j|}{|D_j|},\ \mathrm{R}_{ij} = \frac{|C_i \cap D_j|}{|C_i|},\ \mathrm{F}_{ij} = \frac{2\mathrm{P}_{ij}\mathrm{R}_{ij}}{\mathrm{P}_{ij} + \mathrm{R}_{ij}}$$

各 C_i について，もっとも F-measure が高くなるような D_j を計算し，それを正解集合の C_i における F-measure とする．すなわち，$\mathrm{F}_i = \max_j \mathrm{F}_{ij}$ である．また，このとき，P_i，R_i については，$j' = \mathrm{argmax}_j \mathrm{F}_{ij}$ である j' を用いて，$\mathrm{P}_i = \mathrm{P}_{ij'}$，$\mathrm{R}_i = \mathrm{R}_{ij'}$ として求める．

システムとしての総合評価は，クラスタの要素数を重みとした重みつき平均により与える．よって，システムとしての F-measure(F) は，$\mathrm{F} = \sum_{i=1}^{n} |C_i|\mathrm{F}_i/|\mathcal{C}|$ となる．ただし，$|\mathcal{C}| = \sum_{i=1}^{n} |C_i|$ である．システム全体の Precision(P) および Recall(R) についても同様に計算する．

なお，パラメータは以下のように設定した．近隣文字列マッチングに用いるクエリの前後の単語数 $\alpha = 3$，近隣文字列マッチングの閾値 $\theta_{\mathrm{NSM}} = 1$，NE のマッチングにおいて，人名の共起に対する重み $\beta = 1$，地名の共起に対する重み $\gamma = 0.25$，NE のマッチングの閾値 $\theta_{\mathrm{NE}} = 1$，また階層併合的クラスタリングのカットを行う閾値を $\theta_{\mathrm{VSM}} = 0.0008$ とした．

実験は，人名や地名をクエリとして検索エンジンで検索し，その結果の上位のページ 100 件程度を各クエリについて収集した[4]．収集した各 Web ページについて人手で正解付けを行った上で，それぞれの手法を適用し，各手法の性能を評価した．

実験結果を表 2 に示す．表の中の手法は，VSM がベクトル空間モデルを，NSM が近隣文字列マッチングを，NE が NE を用いたマッチングをそれぞれ表し，2 つ以上の手法が併記されているものは，それらを併用した場合である．最も性能が良いのは NE と NSM を併用，すなわち，NE と NSM の結果のクラスタを併合してクラスタを形成した場合であり，基準（ベクトル空間モデルのみ）に比べて F-measure でおよそ 0.22 の上昇が見られた．

[4] 37 クエリについて計 3859 の Web ページを収集した．

表 2. 実験結果（全 37 クエリの平均）

適用した手法	F	P	R
VSM(Baseline)	0.4596	0.6821	0.5154
NSM	0.5871	0.8302	0.5852
NSM,VSM	0.5510	0.6889	0.6597
NE	0.6311	**0.9484**	0.5585
NE,VSM	0.5874	0.7126	0.6579
NE,NSM	**0.6834**	0.7991	0.7357
NE,NSM,VSM	0.6225	0.6639	**0.7811**

5 終わりに

本発表では，現在開発している名寄せシステムを紹介した．名寄せシステムでは，クエリに関する前提知識を用いることなく，高い精度で実体ごとに文書を分類を行うことを目標とする．また，名寄せの正確さを向上させるための新たな類似度計算法である，近隣文字列を用いたマッチングと NE を用いたマッチングを提案した．Web にある意味の薄いページをジャンクページとして定義し，それらを取り除く仕組みについても説明した．

性能評価の結果，ベクトル空間モデルを用いた方法に対し，近隣文字列のマッチングと NE を用いたマッチングを用いる方法が高い性能を表す（F-measure で 0.2 以上向上している）ことを示した．

参考文献

[1] C. Niu, W. Li and R. K. Srihari: Weakly Supervised Learning for Cross-document Person Name Disambiguation Supported by Information Extraction. *Proceedings of ACL-2004*, pp. 598–605, 2004.

[2] R. Bekkerman and A. MacCallum: Disambiguating Web Appearances of People in a Social Network. *Proceedings of WWW2005*, pp. 463–470, 2005.

[3] B. Larsen and C. Aone: Fast and effective text mining using linear-time document clustering. *Proceedings of the 5th ACM SIGKDD*. pp. 16–22, 1999.