固有表現を対象とした小説登場人物検出

大島一海 [1]　窪田智徳 [1]　小川浩平 [1]　佐藤理史 [1]
[1] 名古屋大学大学院工学研究科
oshima.kazumi.u4@s.mail.nagoya-u.ac.jp

概要

人間は，小説中の登場人物を容易に把握できる．この処理を機械的に行うのが，登場人物検出である．本論文では，固有表現を対象とする登場人物検出器を提案する．本システムは，3 つのモジュールから構成されており，小説テキスト中の人物を指し示す表現（人物表現）を認定し，同一人物を指し示す人物表現に同一の人物 ID を割り当てた後，小説テキスト中の人物表現の出現箇所にタグを付与する．

1　はじめに

人間は，小説を読む際に，小説テキスト中の登場人物を容易に把握することができる．登場人物検出とは，これに相当することを機械的に実現する処理を指す．登場人物を把握することは，小説の理解には不可欠であり，コンピュータによる小説理解の実現のために，解決すべき重要なタスクである．

しかしながら，登場人物検出は，次のような理由により，定義するのが難しいタスクでもある．

1. 小説の登場人物の全集合は，それほど明確ではない．例えば，一瞬しか登場しない人物や名も無い人物（通行人など）も登場人物に含めるのか．
2. 同一人物が異なる表現・言語形式で記述されるため，個々の登場人物をどう記述すべきかが明確ではない．
3. 人物を表す文字列が，人物を指し示さない場合もある．例：「千葉さん」が登場しても，「千葉県」の「千葉」は人物を指し示さない．

これらの点を踏まえ，本研究では，登場人物検出タスクを，以下のようなタグ付けタスクとして定義する．

> 小説のテキスト中に出現する，登場人物を指し示す表現に，人物 ID を属性として持つ人物タグを付与する．同一人物を指す表現には，同一の人物 ID を付与する．主要登場人物[1)]は，必ずタグ付け対象人物に含めることとする．

一般に，人物を指し示す表現（人物表現）には，以下のものがある．

1. 固有表現（名前）
   姓，名，姓名（フルネーム），ニックネーム
2. 固有表現以外
   (a) 人称代名詞
   私，あなた，彼，彼女 など
   (b) 人物を指し示す名詞
   父，母，兄，妹，社長，先生 など

本論文では，これらの人物表現のうち固有表現（名前）をタグ付け対象とする登場人物検出器を提案する．提案する登場人物検出器は，まず固有表現抽出器を利用して人物表現を認定し，次に同一人物判定を行って人物表現に人物 ID を割り当てて辞書を作成し，最後にこの辞書を用いてテキスト中の人物表現に人物タグを付与する．

2　関連研究

2.1　固有表現抽出

登場人物検出は，固有表現抽出 [1] の人物名検出と重なる部分があるが，相違点もある．主な相違点を表 1 に示す．人物名検出の主な対象は実在する人物名であるのに対し，登場人物検出の主な対象は，小説中の架空の人物である．実在しない人物名や突飛な人物名が使用されることもあり，既存の固有表現抽出器をそのまま適用すると，検出漏れや誤検出が起こりやすいと考えられる．一方で，小説では，主要登場人物の人物表現の出現回数がかなり多く，それらのいずれかの出現で人物表現であることが認定できたならば，その事実を利用して，それ以外の出現箇所も正しくタグ付けできる可能性がある．な

1）　物語の中で主要な役割を担うと考えられる登場人物．

表 1　固有表現抽出と登場人物検出の主な相違点

	固有表現抽出	登場人物検出
主な対象テキスト	説明文 （新聞記事 等）	小説
人物の実在性	実在する人物	**架空の人物**
（同一人物の） 出現回数	少ない （高々数回）	主要人物は**多い**
同一人物判定	なし（外付け）	**必須**
検出対象	固有表現	人物表現

お，先に示した登場人物検出のタスク定義においては，同一人物判定が必須である．

2.2　登場人物名検出・抽出

日本語小説の登場人物名検出・抽出に関する研究は，いくつか存在する．馬場ら [2] は，英米文学小説を対象に，人物情報抽出の 1 ステップとして，形態素の品詞と人名辞書を用いて人物名（カタカナ）を抽出している．さらに，姓名・姓・名，および人物名と人物を指し示す名詞の同一人物判定を実現している．一方，米田ら [3] は，主語となっている単語を人物名候補として抽出し，局所出現頻度と述語情報を用いて人物名か否かを判定する手法を提案した．西原ら [4] は，登場人物の関係抽出の前処理として，CaboCha [5] の固有表現解析，日本語語彙大系のカテゴリ情報，選択制約を利用して人物名を抽出している．これら 2 つの手法は，同一人物判定を実現していない．

3　登場人物検出器

3.1　システム構成

提案システムの構成を図 1 に示す．本システムは，人物表現認定（3.2 節），人物表現辞書作成（3.3 節），人物タグ付与（3.4 節）の 3 つのモジュールから構成される．

本システムの入出力は，以下の通りである．

- 入力：小説テキスト（と人物名の読み）
- 出力：人物タグを付与した小説テキスト

作品によっては，人物名にふりがなが付与されていることがある．そのような場合は，人物名の読みを入力として与えてよいこととする．

すでに述べたように，システムのタグ付け対象は固有表現（名前）のみとし，主要登場人物がタグ付け対象人物に漏れなく含まれていればよいこととする．

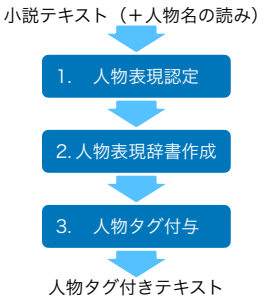

図 1　システム構成

3.2　人物表現認定

固有表現抽出器を用いて人物表現候補を検出した後，その検出結果の誤りを修正または除外することで，人物表現を認定する．人物表現認定は，次のステップで構成される．

1. GiNZA [6] による解析・人物表現候補検出
   オープンソース日本語 NLP ライブラリ GiNZA を用いて，固有表現抽出を実行する．この際，前段処理として，形態素解析および係り受け解析も実行される．固有表現抽出結果のうち，“Person” タグが付与されたスパンを人物表現候補とする．
2. 人物表現候補の修復
   人物表現候補のうち，誤検出である可能性が高いものを修復する．その詳細を付録 A に示す．具体例を，以下に示す．
   - /**千反**//**田**/ → /**千反田**/（連続する人名候補）
   - /**糸魚川**/養子 → /**糸魚川養子**/（名前の組入）
   - /**ち**/ー → /**ちー**/（長音記号の組入）
   - /**折木**/さん → /**折木** | **さん**/（敬称の認識）
   - /**ふくちゃん**/ → /**ふく** | **ちゃん**/（同上）
3. 人物表現候補の適格性判定
   人物表現候補のうち，人物表現である可能性が高いものを適格と判定する．その詳細を付録 B に示す．
4. 人物表現の認定
   適格と判定された人物表現候補のうち，テキスト中に 3 回以上出現しているものを人物表現と認定する．

3.3　人物表現辞書作成

以下の 3 ステップで，人物表現辞書を作成する．

1. 人物表現のタイプ判定
   人物表現（敬称なし）を以下のいずれかのタイ

プに分類する．

- full：フルネーム
- first：名前の先頭部
- last：名前の末尾部[2)]
- reading：名前の読み
- nickname：ニックネーム
- null：上記の 5 タイプに判定できなかった

アルゴリズムを以下に示す．

(a) すべての人物表現に，タイプ null を仮に割り当てる．

(b) first（または last）と full のペアとなる人物表現の組を探す．具体的には，人物表現 X[3)] に対して，X=prefix(Y)（または X=suffix(Y)）を満たす人物表現 Y を探す．ただし，Y のタイプは null か full であり，X と Y の差分は，ひらがな・カタカナ 1 文字ではないことを条件とする．見つかった場合は，X と Y を紐付け，それぞれにタイプを割り当てる．
例 1：X=福部 (first) ↔ Y=福部里志 (full)
例 2：X=里志 (last) ↔ Y=福部里志 (full)

(c) タイプがまだ null の人物表現のうち，ひらがな・カタカナ文字列であるもの (X) を reading と仮定して，読みが X となる人物表現 Y を探す．見つかった場合は，X と Y を紐付け，X にタイプ reading を割り当てる．
例：X=フウカ (reading) ↔ Y=風歌（ふうか）

(d) タイプがまだ null の人物表現のうち，ひらがな・カタカナ文字列であるもの (X) を nickname と仮定し，タイプが reading 以外の人物表現から，読みの類似度が最大の人物表現 Y を選ぶ．類似度は，次式で定義する．

$$\text{類似度} = \frac{\texttt{len}(\text{common prefix})}{\texttt{len}(\text{shorter string})}$$

その類似度が 0.5 以上ならば，X と Y を紐付け，X に nickname を割り当てる．
例：X=ちー (nickname) ↔ Y=千反田（ちたんだ）
(類似度 = len(“ち”)/len(“ちー”) = 0.5)

2. 同一人物判定（人物表現のグルーピング）
各人物表現に人物 ID (PID) を割り当てる．紐付けられた人物表現には，例外を除き，同一の PID を割り当てる．その詳細を付録 C に示す．

3. 人物表現辞書作成
人物表現とその情報を保持する辞書を作成する．辞書のエントリは，以下の 5 項目である．
   - 人物 ID (PID)
   - 人物表現（敬称なし）
   - タイプ
   - 敬称リスト
   - 敬称が必ず付くか（検出結果に基づく）

3.4 人物タグ付与

以下の方法で，人物タグを付与する．なお，文字列検索で用いる人物表現には，敬称なしの人物表現と敬称付きの人物表現の両方を用いる．ただし，敬称が必ず付く人物表現でタイプが full 以外の場合は，敬称なしの人物表現は使用しない．

1. 辞書を用いた人物表現検出
   (a) 文字列検索（最長一致）で，いずれかの人物表現に一致するスパンを見つける．
   (b) 以下の条件を全て満たす場合，そのスパンを検出結果として採用する．
   - 前境界が形態素境界である
   - 後境界が形態素境界であるか，人物表現に敬称が含まれる
   - GiNZA でも検出されている，または，人物表現に敬称が含まれる，または，直後の形態素が人物表現直後の形態素として標準的（付録 D）である
2. タグ付与
検出した人物表現のスパンに人物タグを付与する．タグに含める属性は，人物 ID と敬称（敬称を含む場合）である．

上記の 1(b) により，すべての人物表現の出現に対して人物タグが付与されるわけではない点に注意されたい．例えば，「神山高校」や「千反田家」は，「神山」や「千反田」が人物表現として認定されていても人物タグは付与されない．

4 実験

実装した登場人物検出器を用いて，2 つの実験を行った．GiNZA は，v5.1.3 を使用した．実験に使用した作品の作品 ID と作品名を以下に示す．

1. 谷川流『涼宮ハルヒの憂鬱』
2. 辻村深月『かがみの孤城』
3. 西尾維新『化物語（上）』

2) 日本語名の場合は，first が姓，last が名となる．英語名の場合は，first が given name，last が family name となる．
3) X は文字列が長い順に試す．以降の処理でも同様．

表 2　各手法によるタグ付与の精度

作品	手法	TP	Recall[4)]	Precision	F1
1	GiNZA	1282	0.948	0.987	0.967
	提案手法	1345	0.994	0.985	0.990
2	GiNZA	2788	0.532	0.988	0.692
	提案手法	5163	0.985	0.993	0.989
3	GiNZA	1397	0.449	0.533	0.488
	提案手法	3077	0.989	0.981	0.985

表 3　人物表現の正誤判定結果

作品	正（割合）	誤	計
1	16 (0.80)	4	20
2	43 (0.88)	6	49
3	25 (0.76)	8	33

4.1　実験 1：GiNZA と提案手法の比較

登場人物検出器による検出漏れ・誤検出修正の有効性を確認するため，GiNZA での “Person” タグ付与と，登場人物検出器での人物タグ付与の精度を比較した．タグを付与したスパンが人物を指し示す場合に正解とし，敬称は含んでも含まなくてもよいものとした．結果を表 2 に示す．表中の TP (True Positive) は，正しく検出された箇所数を表す．

いずれの作品においても，提案手法は F1 スコア 0.98 以上を達成し，高い精度で人物タグの付与が可能であることが示唆された．

作品 1 では，GiNZA と提案手法いずれにおいても高い精度となった．

作品 2 では，GiNZA において，Precision は十分に高いものの，主に検出漏れにより Recall が低くなった．一方，提案手法において，Recall が大幅に向上しており，検出漏れを救う処理を担う人物タグ付与モジュールの有効性が確認された．

作品 3 では，GiNZA において，主に誤検出により Recall・Precision ともに低くなった．一方，提案手法において，Recall・Precision ともに大幅に向上しており，誤検出の修正・除外を担う人物表現認定モジュールの有効性が確認された．

4.2　実験 2：同一人物判定の性能評価

まず，人物表現辞書に含まれる人物表現の正誤を人手で判定した．その結果を表 3 に示す．主要登場人物は全て検出されたことを確認した．この表に示すように，提案手法で認定した人物表現のうち，正しいものの割合は，最大でも 0.88 であり，十分に高いとはいえない．この精度は，同一人物判定の性能にも影響することから，人物表現の認定に関しては，さらなる検討が必要である．

4)　真の Recall ではなく，検出した人物表現をもとに算出した疑似的な Recall.

表 4　同一人物判定結果

作品		PID 数		計
		1	2	
1	人物数	9	0	9
2	人物数	18	2	20
3	人物数	13	1	14

表 5　PID ごとの人物タグ付与数

作品	人物	PID_1	PID_2	計
2	A	579	3	582
	B	403	1	404
3	C	968	28	996

次に，正しいと判定された人物表現が，どの人物を指し示すかを人手で判定した後，各人物に付与された PID の種類数を調べた．その結果を表 4 に示す．いずれの作品においても，各人物に割り当てられた PID 数が 1 であった（正しく同一人物判定ができた）割合が高く，同一人物判定は概ね機能していると考えられる．

同一人物に複数の PID が割り当てられたとしても，まれにしか出現しない人物表現の同一人物判定に失敗しているのであれば，それほど大きな問題とはならない．そこで，割り当てられた PID 数が 2 であった人物について，それぞれの人物表現の出現数（人物タグを付与した数）を調査した．その結果を表 5 に示す．このように，同一人物判定に失敗した人物表現の出現数は，3 % (28/996) 以下であるため，大きな問題とはならないと考えられる．

5　今後の課題

現状のシステムには，以下のような課題がある．

1. GiNZA で一度も “Person” タグが付与されなかった人物表現を検出できない
2. 人物表現辞書に人物表現でないものが含まれる

1 つ目の課題に対する解決策として，関連研究 [3, 4] を参考に，主語となっている単語を人物表現候補とする方法や，格フレームの選択制約を利用する方法など，固有表現抽出器以外も利用する検出方法を考えている．

2 つ目の課題に対する解決策として，適格性判定と人物表現認定の増強を考えている．具体的には，関連研究 [3, 4] を参考に，述語情報または格フレームの選択制約を利用した判定の追加や，出現数による人物表現認定の閾値引き上げを考えている．

謝辞

本研究は JSPS 科研費 JP21H03497 の助成を受けたものです.

参考文献

[1] Basra Jehangir, Saravanan Radhakrishnan, and Rahul Agarwal. A survey on Named Entity Recognition — datasets, tools, and methodologies. **Natural Language Processing Journal**, Vol. 3, p. 100017, 2023.

[2] 馬場こづえ, 藤井敦. 小説テキストを対象とした人物情報の抽出と体系化. 言語処理学会第 13 回年次大会発表論文集, Vol. 13, pp. 574–577, 2007.

[3] 米田崇明, 篠崎隆宏, 堀内靖雄, 黒岩眞吾. 述語情報を利用した小説の登場人物の抽出. 言語処理学会第 18 回年次大会発表論文集, Vol. 18, pp. 855–858, 2012.

[4] 西原弘真, 白井清昭. 物語テキストを対象とした登場人物の関係抽出. 言語処理学会第 21 回年次大会発表論文集, Vol. 21, pp. 628–631, 2015.

[5] 工藤拓, 松本裕治. チャンキングの段階適用による日本語係り受け解析. Vol. 43, No. 6, pp. 1834–1842, 2002.

[6] GiNZA - Japanese NLP Library. https://megagonlabs.github.io/ginza/.

A　人物表現候補の修復

人物表現候補の修復では，以下の手順で人物表現候補を修復する．このうち 3 の敬称は，誤検出ではないが，人物表現か否かを判定する際の重要な要素であるため修復する．

1. 隣接する人物表現候補を 1 つの人物表現候補として結合する．
   例：/**千反**//**田**/ → /**千反田**/
2. 直後の形態素が以下の場合は，その形態素を人物表現候補に組み入れる．
   - 品詞が「名詞-固有名詞-人名-名」
     例：/**糸魚川**/養子 → /**糸魚川養子**/
   - 長音記号「ー」
     例：/**ち**/ー → /**ちー**/
3. 直後の形態素または末尾が敬称の場合は，敬称を区別して人物表現候補に組み入れる．
   敬称：さん，ちゃん，先輩，先生 など
   例 1：/**折木**/さん → /**折木 | さん**/
   例 2：/**ふくちゃん**/ → /**ふく | ちゃん**/

B　人物表現候補の適格判定の条件

検出された人物表現候補が以下に示す 3 つの条件のいずれかを満たす場合，適格と判定する．

条件 1　人物表現候補の検出箇所が，以下のいずれかを満たす
- 敬称が付く箇所がある
- 直後が区切り文字（直後なし，または，句読点や括弧，改行）である
- 直後が助詞の箇所が 3 以上ある

条件 2　すでに適格と判定された人物表現候補が suffix となっている

条件 3　以下の全てを満たす
- すでに適格と判定された人物表現候補の部分列でない
- すでに適格と判定された人物表現候補が prefix となっている
- 人物表現候補の文字列検索で一致する箇所が，以下の両方を満たす
  – 10 回以上出現する
  – 条件 1 のいずれかを満たす

なお，文字列 A が文字列 B の suffix となっている (A=suffix(B)) とは，B が A で終わることを表す．同様に，prefix となっている (A=prefix(B)) とは，B が A で始まることを表す．例えば，「里志」は「福部里志」の suffix となっており，「福部」は「福部里志」の prefix となっている．

C　人物 ID の割り当て

それぞれの人物表現に人物 ID (PID) を割り当てるアルゴリズムを以下に示す．

1. full または null に対し，それぞれ異なる PID を割り当てる．
2. first または last に対し，
   (a) full が 1 つだけ紐付けられている場合：
   full と同じ PID を割り当てる．
   (b) full が複数紐付けられている場合：
   出現数最多の人物の PID を割り当てる．
3. reading または nickname に対し，紐付けられている人物表現と同じ PID を割り当てる．

D　人物表現直後の形態素として標準的

形態素が以下のいずれかに当てはまる場合，人物表現直後の形態素として標準的であると定義する．

- 区切り（形態素なし，または，句読点や補助記号，改行）
- 助詞
- 助動詞「だ」「です」
- 副詞，または，副詞可能
- 人物名の直後に出現しやすい語
  例：本人，相手，みたい