3C-6

An Efficient Window-Based Method Using N-gram Indexing for Approximate Entity Extraction

木村 光樹 † 安達 淳 †† 高須 淳宏 ††
† 東京大学大学院 †† 国立情報学研究所

1 はじめに

Approximate Entity Extraction はあるエンティティの集合からなる辞書とテキストが与えられたときに，テキスト中から辞書中のエンティティと類似した部分文字列を検索するタスクである．この問題は，例えば，論文の参考文献のリストが与えられたときに，予め与えられた著者リストを用いて，参考文献情報の中から著者名がどこに記されているかを検索する場合などに用いられる．

一般的な類似文字列検索と違い，Approximate Entity Extraction ではあるエンティティがテキスト中のどの箇所から始まるかが既知でないために，問題は難しい．辞書中のエンティティ全てについてテキストの最初の文字から比較することで解を得ることができるが，扱うデータのサイズが大規模になってきた今日，これは非常に非効率的な手法である．

近年の Approximate Entity Extraction を解くための手法 [1][2] としては，エンティティを短い部分文字列により索引付けを行い，テキスト中でその索引語が見つかれば実際に類似しているかを判定するという手法が一般的である．しかし，この手法では索引語と一つでも一致した箇所について，実際の類似度の計算を行っているために，無駄な類似度計算が生じている．さらに，索引語の保持には複雑な木構造を用いているために索引付けにかかるコストや，索引語の保持にかかる空間計算量が大きくなることも問題である．

そこで筆者らは，索引語構造に配列構造を用いることで索引付けと索引語保持のコストを削減した索引付けの手法を提案する．索引語には長さ N の部分文字列である，N-gram を用い，類似した文字列同士は一定の個数の gram を共有するという事実に着目し，検索窓を設定することで，無駄な類似度計算を減らす手法を提案する．

† Mitsuki Kimura(mick@nii.ac.jp)
†† Jun Adachi(adachi@nii.ac.jp)
†† Atsuhiro Takasu(takasu@nii.ac.jp)
The University of Tokyo (†)
National Institute of Informatics (††)
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo,101-8430 Japan

1.1 表記

T をテキスト，$|T|$ をテキスト長，$T[i]$ を T の $i(1 \leq i \leq |T|)$ 番目の文字，$T[i:j](1 \leq i < j \leq |T|)$ を $T[i]$ から始まり $T[j]$ で終わる T の部分文字列とする．エンティティを e で表し，このエンティティの辞書を D とする $(e_i \in D)$．本稿では類似度には編集距離を用い，ある 2 つの文字列 s, t 間の編集距離を $ed(s,t)$ で表す．

本稿で扱う Approximate Entity Extraction の定義は以下の通りである．

Approximate Entity Extraction

あるエンティティの集合である辞書 D とテキスト T としきい値 K が与えられたときに，$ed(e_i, T[i:i+j]) \leq K$ を満たす，$< e_i, i, j >$ の組を全て見つける

2 提案手法

本稿では，”Approximate Entity Extraction” のための N-gram を用いた索引付け手法と，それを用いた検索手法を提案する．

2.1 N-gram を用いた索引付け

索引付けの手法について述べる．辞書中のエンティティは全て長さの昇順にソート済みであるとする．各エンティティを長さ N の gram に分割し，それを索引語とした転置索引を構築する．このとき転置索引中では，各索引語はエンティティの id とそのエンティティ中の索引語の開始位置を記憶する．また，索引語は全てをエンティティ辞書，検索対象テキストに出現しない特殊文字列を用いて連結し，この連結した文字列の接尾辞配列と接尾辞配列上で隣り合う接尾辞の共通する接頭辞の最長長を収めた，LCP 配列を求めておく．

2.2 検索

検索は次のように行われる．

1. テキストに検索窓を設定する
2. 検索窓中に含まれる索引語を探す
3. 検索窓中でしきい値以上出現しているエンティティ id を保持する

4. 3. で見つかったエンティティidを候補とし，実際の編集距離を計算し，最終的な解を求める
5. 検索窓が検索対象のテキストの終端を含むまで $1-4$ を繰り返す

検索窓の設定

編集距離が k 以内の2つの文字列は，その文字列長の長さの差は k 以内でなければならない．このため，検索窓はこのことを考慮したものとすると効率的に検索を行うことができる．そこで，エンティティ辞書に含まれるエンティティの最長長を e_l としたとき，検索窓長 W は次式により設定する．

$$W = e_l + k \tag{1}$$

テキスト中の類似エンティティ検索は，この窓中で行うものとする．

索引語検索

検索窓中で索引語を検索し，エンティティの出現頻度をカウンタを用意してカウントする．

テキストの検索窓中の文字列を T_w としたとき部分文字列 $T_{w_i} = T_w[i : i+N-1](1 \leq i \leq T_w - N + 1)$ が索引語と等しいかを確認する．このとき T_{w_i} が索引語であれば，転置索引中でこの索引語が持つレコードに含まれる全てのエンティティidのカウントを1増やす．各エンティティの出現頻度カウンタは，出現頻度とともに最後にカウントされた索引語のエンティティ中での出現位置とを記憶しておく．また，同時に索引語の先頭のテキスト中での出現位置を記憶しておく．もし，あるエンティティのカウンタが既に存在していて，新しく見つかった索引語の中でそのエンティティがカウンタの持つ索引語出現位置よりも小さいときには，別のカウンタを作成する．

索引語の出現頻度のしきい値

N-gramを索引語に用いるとき，編集距離が k 以内の2つの文字列 s, t は以下に表す τ 個のgramを最低でも共有する．

$$\tau = \max(|s|, |t|) - N + 1 - N \cdot k \tag{2}$$

このことにより辞書中の各エンティティは以下に示す τ_{e_i} 個のgramが窓内に存在しなければ解候補となりえない．

$$\tau_{e_i} = |e_i| - N + 1 - N \cdot k \tag{3}$$

カウンタが増えたエンティティに関して，カウントが tau_{e_i} 個以上であれば解候補とする．

解の抽出

解候補となったものは，実際に編集距離を計算する．カウンタには，最初に見つかった索引語のテキスト中での位置が一緒に記憶されているので，それをもとにその索引語文字列の前後の編集距離を実際に計算することで，解を抽出する．

窓の更新

検索窓は $i = 1$ から始め，検索対象テキストの部分文字列 $T[i : i+W]$ 中で検索を行う．更新は i の値を1増やすことにより行い，$i + W = |T|$ となったところで検索は終了する．索引語の検索には，接尾辞配列のrank関数を用いる．rank関数 R は，ある接尾辞配列 SA が与えられたときに次式を満たす．

$$SA[R[k]] = k \tag{4}$$

これは，ある文字が接尾辞配列中でどの位置であるかが分かっている場合に，実際の文字列中でその文字がどこに現れるかを返す関数である．このため，N-gramが索引語であるので窓を更新する前の検索窓で行われている文字比較から，このrank関数とLCP配列を用いれば，窓更新のさいの文字比較は1文字分で済む．

また，検索窓を更新するときには，エンティティidの出現頻度の更新を同時に行わなければならない．そのため，窓内で見つかった索引語には，窓内での出現位置を記憶させて保持しておく．窓更新の時には，この出現位置も更新し，窓内に含まれなくなった索引語を索引語とするエンティティidのカウンタを減らすことにより実現できる．

3 おわりに

Approximate Entity Extractionを解くための索引付けと検索手法を提案した．索引付けでは，配列構造を用いることで既存の手法よりも索引付けにかかるコストを削減することができた．また検索手法では，N-gramの特性を活かし，無駄な類似度計算の回数を削減するためにgramの共有数を考慮した検索窓を用いた．

参考文献

[1] W. Wang et al., Efficient approximate entity extraction with edit distance constraints. In SIGMOD, 2009.

[2] Dong Deng et al., An Efficient Trie-based Method for Approximate Entity Extraction with Edit-Distance Constraints. In ICDE, 2011.