ウェブ検索クエリのための 部分一致文字列に対するエンティティ名称予測モデルの提案

豊田 樹生　小松 広弥　熊谷 賢　菅原 晃平
ヤフー株式会社
{itoyota, hkomatsu, kenkumag, ksugawar}@yahoo-corp.jp

1 はじめに

ウェブ検索クエリの多くには部分一致文字列が含まれている．例えば，クエリ “かぐや様” はエンティティ名称 “かぐや様は告らせたい～天才たちの恋愛頭脳戦～” に対する部分一致文字列である[1)]．しかし，検索クエリとして発行された部分一致文字列が，ウェブページに対するクリック回数がわずかか，またはゼロのテイルクエリである場合は，フィードバックを取得することが困難になる．こういったテイルクエリに対してもエンティティリンキングを適切に行えるようにすることは検索性能の高いシステムを作るうえでとても重要である．

そこで，本研究では，エンティティリンキングの構成要素のひとつとして，ウェブ検索クエリのための部分一致文字列に対するエンティティ名称予測モデルを提案し，次のような貢献を行う:

(i)ブロッキング [8] を用いた大規模分散処理のための効率的な訓練事例の自動生成方法を提案する．
(ii)名称予測モデルのための新たな素性を提案する．
(iii)フィードバックの取得できない事例に対する PU(Positive Unlabelled) 学習 [4] を用いたラベリング手法を提案する．
(iv)比較実験において，Random Forest[2] の予測値とクリック頻度に基づくモデルの予測値を線形補間により組み合わせることで，nDCG@5 の観点で高い性能を達成したことを示す．

2 問題定義

クエリ q 中の主要語 s_s，周辺語 s_c が与えられたときにエンティティの名称の候補のランキング $\langle s_1, ..., s_n \rangle$ を算出する．

各々の s_k(for $k = 1...n$) は次のようにスコアリングされる:

$$score(s_k) = \max\{\psi(s_k, s_s, s_c, e) | e \in E, s_k \in S_e\}$$

ここで，ψ はスコアリング関数，E は知識ベース内でのエンティティの集合，S_e はエンティティ e から展開できるエンティティ名称の候補の集合である．

3 提案手法

本研究は，ブロッキングを用いた訓練事例の生成 (3.1)，PU 学習 (3.2) の二つから構成されている．

3.1 訓練事例の生成

正例の取得　次の条件をすべて満たした場合，クエリ q とエンティティ名称 s は部分一致していると判定し，クエリ q 中の主要語 s_s，周辺語 s_c，および，エンティティ名称 s の組を正例とする:

(i)主要語 s_s とエンティティ名称 s が pkduck[13] の判定式 $|s_s| - |LCS(s_s, s)| \leq \delta$ を満たす ($\delta = 0$)．あるいは事前に辞書登録されている．
(ii)q 中の周辺語 s_c がエンティティ e の許可リストに登録されている[2)]
(iii)エンティティ名称 s と対応するウェブページに対してクエリ q でクリックがある．

ラベル未付与の事例の取得　正例と判定された事例から q-gram[5] を基にした手法によりブロッキングのためのキーを抽出する．手順は次のとおりである: 1) エンティティ名称 s を通常の q-gram と同様に文字 n-gram に分割する．例えば n=2 のとき “伊藤健太郎” は “伊藤”,“藤健”,“健太”,“太郎” の 4 つのキー (主キー) に分割される．2) クエリ q 中の周辺語 s_c(サブキー)[3)] と主要語から抽出されたキーを結合

1） “かぐや様” に対する候補は複数考えられるが，ここでは 2020 年 12 月現在の検索意図を考慮している

2） この許可リストはエンティティ e と対応する概要文に周辺語 s_c を照合するなどの操作により自動的に生成される．

3） 周辺語が欠損している場合は周辺語モデルを参照し確率最

する．例えば n=2，s_c=“声優” のとき，“伊藤_声優”,“藤健_声優”,“健太_声優”,“太郎_声優” という 4 つのキーが生成される．3) 結合後のキーの生起頻度を計測し，一定頻度を超えるキーは除外する．

ラベル未付与の事例に対しても同様の方法でブロッキングのためのキーを抽出する．ここで，ラベル未付与の事例は，周辺語モデルおよび名前エンティティモデル [17] のエンティティ ID 同士を結合したレコードから生成される．このレコードのエンティティ名称および周辺語からそれぞれ主キー，サブキーを生成する．全ての事例を対象として，同一のキーを持つもの同士でブロックを構成する．ブロックの要素がラベル未付与事例のみで構成されている場合はそのブロックを除外する．ラベル未付与の事例に対する主要語 s_s を同ブロック内の正例の主要語を複製することで生成する．[4] 最後に残った事例を学習に用いる．

素性の抽出　正例およびラベル未付与の事例それぞれに対し，クエリ素性 (Q)，エンティティ素性 (E)，クエリエンティティ素性 (QE) の 3 種類の素性の抽出を行った．素性の一覧を表 1 に示す．

Randomized SVD　本研究では素性のうちの一部を Randomized SVD[3, 7] により次元圧縮したうえで Random Forest への入力とする:

$$X = U\Sigma V^* \tag{1}$$

$$XV = U\Sigma \tag{2}$$

ここで X は各行が素性のベクトルである行列，U,V は回転行列，Σ は特異値の対角行列である．X に対して次元圧縮の回転行列 V を適用することは $V\Sigma$ を求めることと等価であるため，これを求めることで次元圧縮を行う．

3.2 PU 学習

本研究では次のように PU 学習における Double Weighting[4] を適用する:

1) 正例およびラベル未付与の事例を訓練用とテスト用の 2 つに分割する．2) 訓練用の正例およびラベル未付与の事例を入力とし，ラベルが付与されているか否かの予測器 A を生成する．3) テスト用の正例に対して予測器 A を適用し，ラベル付与確率 $g(x)$ の平均 c を求める．4) テスト用のラベル未付与の事例に対して予測器 A を適用し $w(x) = p(y = 1|x, s = 0)$ の重みによりラベリングを行う．ここで $w(x)$ は定数 c への依存を持つ．5) テスト用事例と訓練用事例を入れ替え，2~4 のステップを行う．6) 全ての事例にラベルが付与されたことを利用し，エンティティ名称の生成確率の予測器 B を生成する．

4 評価

4.1 比較手法

DM　クエリ補完モデル [17][5]．クリックログを用いて所与のクエリに対するエンティティ名称候補の生起確率をディリクレ-多項分布により表現したモデル．遷移先と遷移元のクエリが同一になりやすいか否かによりパラメータベクトルを設定．候補には遷移元のクエリの複製も含まれる．

CLK　DM からクエリの複製操作を取り除き，多項分布化した (パラメータベクトルのすべての要素をゼロにした) モデル

RF　Random Forest[2]．Spark MLlib 2.4.5 を使用．SVD による次元圧縮には Criteo Spark-RSVD[6] を使用．

COMB(RF+CLK)　RF と CLK の予測値の線形補間モデル: $\psi_{score} = \beta * \psi_{CLK} + (1 - \beta) * \psi_{RF}$

COMB(RF+DM)　RF と DM の予測値の線形補間モデル: $\psi_{score} = \beta * \psi_{DM} + (1 - \beta) * \psi_{RF}$

4.2 データセット

クリックログ　2019 年 12 月 14 日から 2020 年 12 月 14 日の 1 年間にヤフー検索に対して発行されたクエリ，および，そのとび先 URL のログ．

タクソノミーおよび知識ベース　2020 年 12 月 15 日付けの内製の統合的知識ベース [15].

訓練事例　3.1 に示した手順により訓練事例の作成を行った．正例 8,600,161 件，ラベル未付与事例 4,773,930 件が生成された．

4.3 評価用事例の作成

次の手順により評価用事例を作成した: 1) RF と CLK の予測結果間で順位一位のエンティティ名称が異なるようなクエリを抽出する[7] 2) 周辺語の付与されたクエリのうち人物，建造物，漫画，映画カテゴリに属するクエリをそれぞれ 25 例ランダムサン

大のエントリを用いて補完

4) このとき pkduck の判定式を用いて部分一致の制約を満たすか否かの検査を行う．条件を満たさない場合は複製しない．

5) 実験条件をそろえるため，集合 $S_{(ss,sc)}$ に要素を加える条件は pkduck の判定式を満たすか否かに変更した

6) https://github.com/criteo/Spark-RSVD

7) 一位の異なるクエリは 551,039 例，同一のクエリは 7,084,003 例

プリングして抽出する 3) 周辺語の付与されていないクエリのうち空白文字を含むクエリ，含まないクエリをそれぞれ 25 例ランダムサンプリングして抽出する 4) 計 150 クエリをヤフー検索に対して発行する[8]．返却されたウェブページのそれぞれに対して，対応するエンティティ名称を記録する[9]．5) クエリ-エンティティ名称に対して，3 スケールで評価値を付与する: 所与のエンティティ名称が検索結果の 1 位[10]，もしくは上位 10 件の 50% 以上を占めている場合は 2，それ以外で検索結果に含まれる場合は 1，検索結果に含まれない場合は 0 とする．

4.4 nDCG@5 の比較

4.3 で作成された評価用事例を用いて nDCG@5 の計測を行った．図 1 に線形補間の係数 β を [0,1] の範囲で変えたときの値の変化を示す．β の値が $[0, 0.25)$ の範囲では CLK が提案手法の COMB(RF+CLK) を上回る 0.568 を示した．β の値が [0.25,1.0] の範囲では提案手法の COMB(RF+CLK) が CLK を上回る 0.593 を示した[11]．表 2 に COMB(RF+CLK) が nDCG@5 の改善に貢献した実例を示す．一方，DM は比較手法間で最も低い性能となった．DM は遷移元のクエリを複製し，エンティティ名称の候補に加えるという操作が含まれている．そのため，1 位のエンティティ名称が正答と一致しにくくなり，nDCG のように順位の影響を受けやすい評価指標において不利に働いたのではないかと考えられる．

4.5 素性の分析

図 2 に Random Forest の出力したジニ不純度に基づく素性の重要度の上位 20 件を示す．

全体で最も効果が高かったのは *MinMaxClickFreq* で，次点は *MinMaxContProb* であった．一方，最小最大スケーリングを行わなかった *ClickFreq* および *ContProb* は上位 20 件内ではあるものの効果は大幅に劣るという結果になった．絶対値ではなくエンティティ名称候補間での相対値が重要であるということがわかった．

全体で三番目に効果が高かったのは *GrubbsSmirnov* であった．訓練事例の中に，あるクエリに対する候補エンティティ名称間での *ClickFreq* の分布を考えた時に，外れ値を含むような分布が多く含まれていたということが考えられる．

Randomized SVD により次元圧縮を行った *ClassVec*，*SfreqVec*，*ContLocVec* の 3 つのベクトルに関しては，この 3 つの中では相対的に *ContLocVec* の効果が最も高かった．*ContLocVec* の分散表現は，エンティティのクラスや周辺語を *ClassVec* および *SfreqVec* よりも細かな粒度で表現しているため，これら 2 素性の役割を包含しているのではないかということが考えられる．

また *ContLocVec* 自身よりもそこから派生した *ContLocVecCosMean*，*ContLocVecCosVar* の方が効果が高かった．ベクトル間の関係性を要約統計量で表現することで，より効果が高まったのではないかと考えられる．

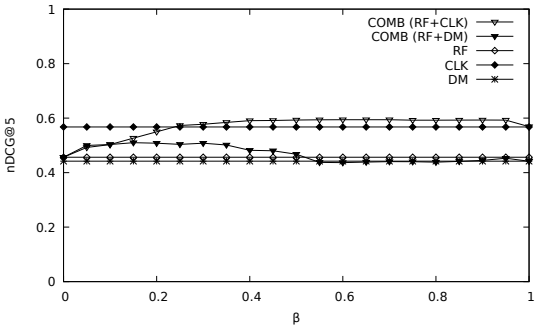

図 1: nDCG@5 の比較．RF，CLK，DM は β とは非依存である

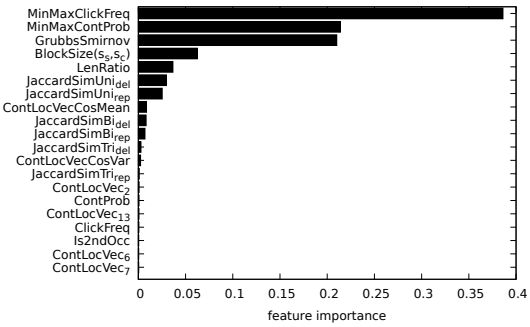

図 2: Random Forest の出力した素性の重要度．*ContLocVec* に対する添え字はベクトルの要素の番号

5 おわりに

本研究ではエンティティ名称の予測モデルの提案を行った．今後の課題としては，より大規模な評価を行うということやタイプクエリ判定 [10] や人物の本業判定 [12] などの関連分野の技術をうまく組み合わせるということがあげられる．

8） 2021 年 1 月 5 日の検索結果を利用
9） 対応が存在しない場合は EMPTY というメタタグを付与
10） EMPTY の場合は空位
11） Wilcoxon-Pratt signed-rank test[11, 14], $p < 0.05$

参考文献

[1]Roi Blanco, Giuseppe Ottaviano, and Edgar Meij. Fast and space-efficient entity linking for queries. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, pp. 179–188. ACM, 2015.

[2]Leo Breiman. Random forests. *Machine learning*, Vol. 45, No. 1, pp. 5–32, 2001.

[3]Paul G Constantine and David F Gleich. Tall and skinny qr factorizations in mapreduce architectures. In *Proceedings of the second international workshop on MapReduce and its applications*, pp. 43–50, 2011.

[4]Charles Elkan and Keith Noto. Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 213–220, 2008.

[5]Luis Gravano, Panagiotis G Ipeirotis, Hosagrahar Visvesvaraya Jagadish, Nick Koudas, Shanmugauelayut Muthukrishnan, Divesh Srivastava, et al. Approximate string joins in a database (almost) for free. In *VLDB*, Vol. 1, pp. 491–500, 2001.

[6]Frank E Grubbs. Procedures for detecting outlying observations in samples. *Technometrics*, Vol. 11, No. 1, pp. 1–21, 1969.

[7]Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, Vol. 53, No. 2, pp. 217–288, 2011.

[8]George Papadakis, Dimitrios Skoutas, Emmanouil Thanos, and Themis Palpanas. Blocking and filtering techniques for entity resolution: A survey. *ACM Computing Surveys (CSUR)*, Vol. 53, No. 2, pp. 1–42, 2020.

[9]Marius Paşca, Benjamin Van Durme, and Nikesh Garera. The role of documents vs. queries in extracting class attributes from text. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pp. 485–494. ACM, 2007.

[10]Ellie Pavlick and Marius Pasca. Identifying 1950s american jazz musicians: Fine-grained isa extraction via modifier composition. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2099–2109, 2017.

[11]John W Pratt. Remarks on zeros and ties in the wilcoxon signed rank procedures. *Journal of the American Statistical Association*, Vol. 54, No. 287, pp. 655–667, 1959.

[12]Mahsa S Shahshahani, Faegheh Hasibi, Hamed Zamani, and Azadeh Shakery. Towards a unified supervised approach for ranking triples of type-like relations. In *European Conference on Information Retrieval*, pp. 707–714. Springer, 2018.

[13]Wenbo Tao, Dong Deng, and Michael Stonebraker. Approximate string joins with abbreviations. *Proceedings of the VLDB Endowment*, Vol. 11, No. 1, 2017.

[14]Frank Wilcoxon and Roberta A Wilcox. *Some rapid approximate statistical procedures*. Lederle Laboratories, 1964.

[15]Tomoya Yamazaki, Kentaro Nishi, Takuya Makabe, Mei Sasaki, Chihiro Nishimoto, Hiroki Iwasawa, Masaki Noguchi, and Yukihiro Tagami. A scalable and plug-in based system to construct a production-level knowledge base. In *Proceedings of the 1st International Workshop on Challenges and Experiences from Data Integration to Knowledge Graphs co-located with the 25th ACM SIGKDD International Conference on Knowledge Discovery &Data Mining*, 2019.

[16]豊田樹生, 土沢誉太, 築地毅, 菅原晃平, 野口正樹. dishpam: A distributable seeded hierarchical pachinko allocation model. 言語処理学会第 26 回年次大会発表論文集, pp. P2–19, 2020.

[17]豊田樹生, 夜久真也, 石川葉子, 土沢誉太, Kulkarni Kaustubh, Bhattacharjee Anupam, 宰川潤二. ウェブ検索クエリに対する周辺語を考慮した教師なしエンティティリンキング. 言語処理学会第 25 回年次大会発表論文集, pp. 81–84, 2019.

表 1: 素性の一覧．クエリ素性 (Q)，エンティティ素性 (E)，クエリエンティティ素性 (QE)

素性	説明	種類
$NumToken(q)$	クエリ q の空白区切りでのトークン数	Q
$Length(s_s)$	主要語 s_s の文字数	Q
$BlockSize(s_s, s_c)$	主キーが主要語 s_s と一致し，サブキーが周辺語 s_c と一致しているブロックのサイズ	Q
$BlockSize(s_c, s_s)$	主キーが周辺語 s_c と一致し，サブキーが主要語 s_s と一致しているブロックのサイズ	Q
$BlockSize(s'_s, s''_s)$	主要語 s_s を接頭辞 s'_s, 接尾辞 s''_s に分割したときに主キーが s'_s，サブキーが s''_s に一致するようなブロックのサイズ．複数候補がある場合は最大のサイズのブロックを選択．	Q
$IsFilledCont(s_c)$	欠損値補完により補われた周辺語 s_c か否か	Q
$CatNameFreq(s_s, s_c)$	主要語 s_s，周辺語 s_c を結合した文字列と一致しているエンティティ名称の数	Q
$SfreqVec(s_c)$	クラス内での周辺語 s_c の Sfreq[9] の分布を周辺語-クラス方向に変換したベクトル (136 次元) を Randomized SVD で 15 次元に圧縮したもの	Q
$ClickFreq(e)$	クエリ非依存でのエンティティ e と対応するウェブページに対するクリック回数 (底 10 で対数変換)	E
$ClassVec(e)$	エンティティ e の属するクラスのベクトル (136 次元) を Randomized SVD で 15 次元に圧縮したもの	E
$Length(s)$	エンティティ名称 s の文字数	E
$GrubbsSmirnov(e, q)$	Grubbs-Smirnov 棄却検定 [6] の統計量 $T = \frac{X_{(n)} - \mu}{\sigma}$ をクエリ q に対するエンティティ名称候補で最大の $ClickFreq(e)$ を持つ事例に対し計測したもの．ここで $X_{(n)}$ は最大値．σ，μ はそれぞれ最大値を除いて計算した標準偏差および平均	QE
$MinMaxClickFreq(e, q)$	$ClickFreq(e)$ をクエリに対するエンティティ名称候補内で最小最大スケーリングした値	QE
$EntityProb(s, e)$	エンティティ名称 s が所与のときのエンティティ e の生成確率 (FEL [1])	QE
$ContProb(e, s_c)$	エンティティ e が所与のときの周辺語 s_c の生起確率 [16]	QE
$MinMaxContProb(e, s_c, q)$	$ContProb(e, s_c)$ をクエリに対するエンティティ名称候補内で最小最大スケーリングした値	QE
$ContLocVec(e, s_c)$	エンティティ e が所与のときの周辺語 s_c の知識グラフ上で生起している場所のベクトル (417 次元) を Randomized SVD で 15 次元に圧縮したもの	QE
$ContLocVecCosMean(e, s_c, q)$	$ContLocVec(e, s_c)$ をクエリに対する候補集合ごとにコサイン類似度の平均をとったもの	QE
$ContLocVecCosVar(e, s_c, q)$	$ContLocVec(e, s_c)$ をクエリに対する候補集合ごとにコサイン類似度の分散をとったもの	QE
$LenRatio(s_s, s)$	主要語 s_s とエンティティ名称 s の文字数の比率: $LENGTH(s_s)/LENGTH(s)$	QE
$JaccardSimTri(s_s, s)$	主要語 s_s, エンティティ名称 s 間の文字 (2 文字)tri-gram でのジャッカード類似度．対象文字列に対してそのまま，記号置換，記号削除の 3 パターンの処理を行う．	QE
$JaccardSimBi(s_s, s)$	主要語 s_s, エンティティ名称 s 間の文字 (2 文字)bi-gram でのジャッカード類似度．対象文字列に対してそのまま，記号置換，記号削除の 3 パターンの処理を行う．	QE
$JaccardSimUni(s_s, s)$	主要語 s_s, エンティティ名称 s 間の文字 (2 文字)uni-gram でのジャッカード類似度．対象文字列に対してそのまま，記号置換，記号削除の 3 パターンの処理を行う．	QE
$Is2ndOcc(s_c, e)$	周辺語 s_c がエンティティ e の持つ occupation 要素でのみ生起しているか否か	QE

表 2: 実際の出力例．スコア順にエンティティ名称を列挙した．評価点 2 の事例は太字にした．正答と一致するエンティティ名称には下線を引いた．

クエリ	正答	CLK	COMB(RF+CLK)
阿部 声優	**阿部敦**, 阿部大樹, 阿部玲子	**阿部敦**	**阿部敦**, 阿部寛,阿部玲子, 阿部信行, 阿部六郎
アフリカ 動物園	いしかわ動物園, 京都市動物園, 天王寺動物園, アフリカンサファリ, 九州動物自然公園, よこはま動物園, 南アフリカ国立動物園	南アフリカ国立動物園	南アフリカ国立動物園,アフリカンサファリ
枚方 公園	**枚方パーク**, 枚方公園駅, 淀川河川公園, 山田池公園	枚方市立　王仁公園	枚方市立　王仁公園,**枚方パーク**, 枚方駅, 枚方市立陸上競技場, 枚方市立山田中学校