機械学習を利用した質問応答型検索システムの自動構築機構の実装

島崎　祐輔† 　横山　昌平†† 　福田　直樹†† 　石川　博††

† 静岡大学大学院情報学研究科　〒 432–8011 静岡県浜松市中区城北 3-5-1
†† 静岡大学情報学部　〒 432–8011 静岡県浜松市中区城北 3-5-1
E-mail: †gs08033@s.inf.shizuoka.ac.jp, ††{yokoyama,fukuta,ishikawa}@inf.shizuoka.ac.jp

あらまし　既存の質問応答型システムは，1 つの質問応答型システムで様々な種類の質問に対して対応できるようにすることを目標としている．一方で，多くの種類の質問を対象とすることで，システムの構成が複雑になったり，質問に対する解答の品質を高く保つことが難しいなどの課題が生じる．そこで，ある特定の種類の質問に対して特化した Web 質問応答型検索システムを，検索アプリケーションとして目的ごとに用意することを考える．プログラミング経験のないユーザが，検索エンジンの結果を利用し自分の目的にあった検索アプリケーションを作りたいと考えた場合，実際にプログラミングをして独自の検索アプリケーションを構築することは容易ではない．本論文では，プログラミング経験のないユーザでも Web 上の操作のみで作成可能な質問応答型検索システムの自動構築機構の実装について述べる．

キーワード　情報検索，機械学習

Machine Learning Based on Automated Composition Generator for Web QA-Search Systems

Yusuke SHIMAZAKI†, Shohei YOKOYAMA††, Naoki FUKUTA††, and Hiroshi ISHIKAWA††

† Graduate School of Informatics, Shizuoka University　Johoku 3–5–1, Naka-ku, Hamamatsu-shi, Shizuoka, 432–8011 Japan
†† Department of Computer Science, Faculty of Informatics, Shizuoka University　Johoku 3–5–1, Naka-ku, Hamamatsu-shi, Shizuoka, 432–8011 Japan
E-mail: †gs08033@s.inf.shizuoka.ac.jp, ††{yokoyama,fukuta,ishikawa}@inf.shizuoka.ac.jp

Abstract　Ordinary QA-systems aim to cover various questions by a single system. By having wide range of target questions to be answered by a single system,the architecture of those systems are complicated and makes it difficult to keep the quality of answers for such questions. In our approach, we generate QA-systems each of which covers only a specific type of questions. Our system enables users to generate their own QA-systems from results of search engines, even when they do not have enough programming knowledge. In this paper, we implement automated composition mechanism for web QA-search systems that can be used by presenting some examples and heuristics to the system on the web.

Key words　information retrieval，machine learning

1. ま え が き

検索エンジンは，Web 上の大規模データとユーザをつなぐインタフェースとして使われている [1]．ユーザの様々な要求に応えるために，検索エンジンの高精度化のみでなく，通貨換算や列車乗り換え検索との連携など，単なる Web 検索の枠をはみ出した検索目的とのリンクに関する研究および実現化が行われてきている．一方で，それらの試みは，開発上のコストの兼ね合いから，ある程度のユーザ数が見込めるような一般的な目的のものが優先されてきたと考えられる．

また，検索エンジンの利用者にはプログラミング経験のないユーザも多いと考えられる．iGoogle のようにいくつかのコンポーネントを組み合わせて自分好みの Web アプリケーションを構築する動きも広がってきている．例えば，プログラミング経験のないユーザが，検索エンジンを利用して有名人の俗称を検索するようなアプリケーションを作りたいと考えたとする．

松井秀喜というキーワードで検索して結果のサマリ内に「ゴジラこと松井秀喜」という文章があったときに，ユーザは“ 松井秀喜 ”と単語間の距離が近い単語に答えがあるのではないかと推測することはできる．

しかしながら，プログラミング経験のないユーザには，そのような推測をすることができても，実際にプログラミングをしてそのような処理を行うアプリケーションを構築することは容易ではない．本研究では，複雑なプログラミングを行うことなく，Web 上での簡単な操作により，個々の目的にあった検索アプリケーションを構築できるようにすることを考える．

2. 質問応答型検索システム

2.1 質問応答型検索システムの利点

検索語に基づく Web 検索では，その検索語に関係するページの提示を目的としており，特定の問題に対する答えそのものは提示されない．また，複数の検索語を同時に使用した際には，その検索語間の関係を明示しにくいなどの課題がある．これらの点が問題となるような場合には，ユーザの検索要求を質問文という形で与え，それに対する回答そのものを提示するというアプローチもあり得る．そのアプローチの一つとして，Web 上での質問応答システムがある．

Web 上での質問応答システムには，質問への答えを他のユーザに求める人力型のシステム（人力質問応答システム）と，自然言語解析技術と検索エンジンから返される結果からの統計的な情報を組み合わせた，自動応答作成型のシステム（Web 質問応答型検索システム）がある．Yahoo!知恵袋 [2] や教えて！goo [3] など，あるユーザの質問に対して参加者同士で知恵や知識を出し合い解決していくサービスは人力質問応答システムである．人力質問応答システムのメリットには，質問文の解釈や解答が人手で行われ，質問を解析する処理や答えを大量の文章から機械的に抽出する処理の必要がないため，複雑な情報抽出技術の適用が不要な点がある．一方で，Web 質問応答型検索システムのメリットには，人手では処理しきれない膨大な量の質問を機械的に扱える点である．本論文で扱う質問応答システムは，後者の Web 質問応答型検索システムである．

Web 上でのユーザの（QA サイトにおける）質問は,「情報検索型」と「社会調査型」の 2 つに大別することができる [4]. 情報検索型の質問は，サーチエンジンや図書館のレファレンス・サービスを利用して回答を探すことが可能な質問である．一方，社会調査型の質問は，客観的な正解はなく，特定の個人あるいは集団に対してアンケート調査を行うことで回答を得るような質問である．一般に，質問応答型検索システムには，情報検索型の質問に多く見られる質問・回答の内容表現の類似度に基づいて機械的に回答を抽出するアプローチが有効であると考えられる．

一般に，Web 質問応答型検索システムは，3 つの主要なコンポーネントで構成されている [5]．1 つ目は，検索クエリを扱う情報抽出エンジンである．Web 上のコンテキストの場合，これは Web ページを索引する検索エンジンが担っている．2 つ目は，自然言語の質問を情報抽出エンジンに対するクエリに変えるためのクエリ形成方法である．これは,（文書）コレクションから関係のある（質問の答えを含んでいそうな）文書を抽出するために行われる．3 つ目は，文書を分析し文書から答えを抽出する解答抽出である．

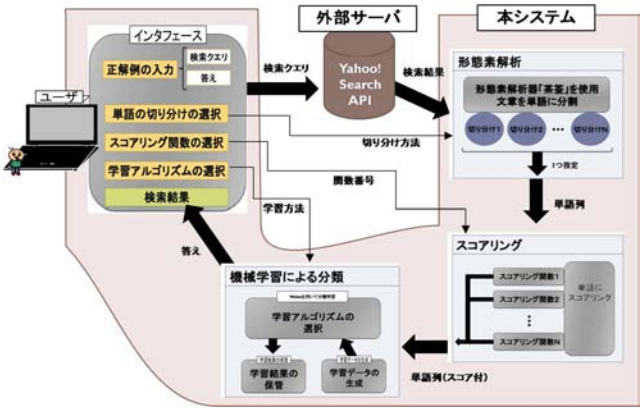

図 1　システムの構成

Web 質問応答型検索システムを構築する 1 つのアプローチとして，機械学習を用いてシステムを構成する方法がある [6] [7]. 例えば，質問文の解析（ユーザの質問文から答えのタイプの同定）や解答抽出・選択部分に分類学習を用いたアプローチが提案されている [6]．また，質問文の解析などの各モジュールに対して個々に機械学習を適用するのではなく，すべてのモジュールをまとめて学習させる手法も提案されている [7].

2.2 本研究での質問応答型検索システム

ユーザの質問に Web 検索結果を用いて解答するには，次の方法がある．例えば,「Who was the first American in space?」という質問の場合には,「the first American in space」というフレーズで検索を行い，検索結果から文法的な構造に着目し「the first American in space was 人物名」という文章を探し，文章から人物名を抽出して質問者に人物名を提示することが考えられる．また，出現頻度などの Web 上の統計的な情報に着目して,「the first American in space」というフレーズで検索を行い，検索結果から出現頻度の高い名詞（人名）などを質問者に提示する方法もある．質問が難しい場合には，今述べた 2 つの手段を組み合わせて解決することも考えられる．

この種の質問応答型システムは，1 つの質問応答型システムで様々な種類の質問に対して対応できるようにすることを目標としている．一方で，多くの種類の質問を対象とすることで，システムの構成が複雑になったり，質問に対する解答の品質を高く保つことが難しいなどの課題が生じる [8].

本研究では，ある特定の種類の質問に対して特化した質問応答型システムを，目的ごとに用意することを考える．例えば,「リスを見ることのできる動物園はどこか？」というような質問文を入力として与えるのではなく，動物名を入力したらその動物を見ることのできる動物園の一覧を表示するような質問応答型システムである．

3. 正例からの検索システムの自動構築

3.1 本システムの概要

本研究では，質問応答型検索システムを作成するための自動構築機構を実装した．自動構築機構を利用することによりプログラミングをしたことがない人でも，そのような質問応答型システムを容易に作成できるようにする．本研究では，ある目的のための質問応答型検索システムを作成するユーザは，質問文を入力する代わりに，検索エンジンへのクエリと答えを入力する．たとえば，先ほどの動物園を探すことに特化した質問応答型検索システムを作成しようとする場合には検索エンジンのクエリとして ”リス ”とリスが見られる動物園の ”井の頭動物園，東山動物園，etc… ”を入力する．このように質問文ではなく，検索エンジンへのクエリと答えを自動構築機構に与えることで特定用途に特化した質問応答型システム（以降，単にアプリケーションと呼ぶ）の生成を行う．図 1 に本システムの構成を示す．検索エンジンには，Yahoo! デベロッパーネットワークが提供する検索 Web API [9] を利用した．説明の例題として，動物名から，その動物を見られる動物園を探す検索アプリケーションを考える．

例えば，動物名「リス」を見ることのできる複数の動物園を Web 検索結果から得る方法として，次のようなヒューリスティックスに基づく方法がある．まず,「リス　動物園」を検索し検索結果のサマリを得る．サマリより動物園を含む文字列を抽出する．Web 上にある膨大な量の文書による統計的性質を利用して，正解となる単語列（「井の頭動物園，王子動物園，etc…」）を抽出し，ユーザに提示する．このようなヒューリスティックスに基づく検索であっても，実際には調整が必要なパラメータが多くある．本研究では，形態素解析による語の粒度の指定やスコアリング関数の選択などのパラメータに調整が必要であり，そのパラメータ調整部分はアプリケーションを作成するユーザが行う．答えの抽出には機械学習を用いた．

3.2 正例からの学習による答えの自動抽出

先ほど例題として示した，動物名からその動物を見られる動物園を検索するアプリケーション（以降はこれをアプリケーション A と呼ぶ）の自動構築について，その具体的な構築手順を示す．動物園を検索するアプリケーションを構成しようとするユーザは，正例の入力，単語の切り分けの選択，スコアリング関数の選択，学習アルゴリズムの選択を行う．

正例の入力では，あらかじめ用意した検索語と正答をシステムに入力する．検索語としては「ゴリラ・クジャク・サイ」などの動物名を入力する．正答は，検索語が“ ゴリラ ”ならば,“ ゴリラ ”を見られる動物園の名前とする．また，アプリケーションが意図した性能を発揮できるようにするための学習データとするためには，適切な数の正例が必要になると考えられる．

正答候補となる語の切り分けには，形態素解析器を使用して文章から切り出した形態素に対して，語の品詞やカタカナ文字列のみを扱うなどの語の属性の指定などを行う．アプリケーション A では，答えとして「井の頭動物園」などの名詞が返ってきてほしいので，正答候補語の品詞は名詞のみを扱うように

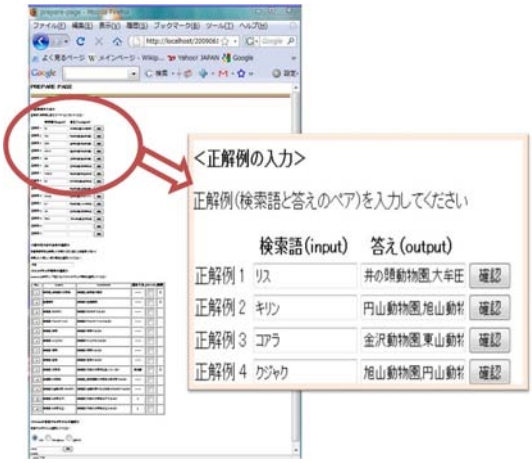

図 2　検索アプリケーション作成画面

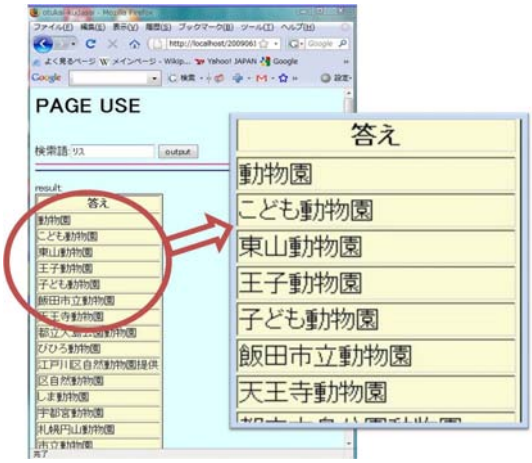

図 3　検索アプリケーション実行画面

選択したりする．また，アプリケーション A とは関係ないが，答えとして「カタカナ」の文字列が返ってきてほしいなど正答に用いられる文字があらかじめ分かっている場合には，その指定を言葉の種類選択で行うこともできる．

スコアリング関数の選択では，システムで事前に用意してある 13 種類（表 1）から適切なものをいくつか選ぶ．例えば，アプリケーション A の場合は，関数番号 1・関数番号 2・関数番号 9 を使用し，関数番号 9 のある対象の文字列には「動物園」とする．

アプリケーション A を作成するユーザは学習アルゴリズムを選択することができる．

以上 4 つのパラメータを決定し作成ボタンを押下すると図 3 の結果出力画面に遷移する．検索ボックスに単語を入れて，検索ボックス右側のボタンを押下すると実行結果が出力される．

3.3 解答候補のランキング

本自動構築機構で生成されたアプリケーションは，質問の解答となる候補語それぞれに対して，機械学習に基づいて正答であるかを判定する．そのため，本来はその質問の解答ただ 1 つである場合であっても，複数の解答が提示される場合がある．すなわち，アプリケーションによって得られた答えが複数ある場合に，その答えをランキングして使用するユーザに提示する

表 1　スコアリング関数の一覧

関数名	概要
関数 1	候補語と検索語の関係
関数 2	候補語の出現頻度
関数 3	候補語がカタカナであるか
関数 4	候補語がアルファベットであるか
関数 5	候補語が数字であるか
関数 6	候補語がひらがなであるか
関数 7	候補語が漢字であるか
関数 8	候補語が記号であるか
関数 9	候補語がある対象の文字列を含んでいるか
関数 10	候補語間の文字数は何文字であるか
関数 11	候補語の先頭文字が対象のカタカナであるか
関数 12	候補語がある文字数以下であるか
関数 13	候補語がある文字数以上であるか

必要がある．本研究では，正答が一つのみ存在し得られた答え (以降はこれを答え候補と呼ぶ) が複数ある状況に対して焦点を当て，4 つのランキングを実装した．

1 つ目は，次のようなヒューリスティックスを適用した．答え候補に数多く出現する単語は，重要な答えである可能性がある．そこでランキングは，スコア（答え候補の中での出現回数）の高い順にランキングされる．このスコアが高いほど上位にランキングされる．

2 つ目は,「候補語の検索のヒット件数（件数 A)」と「候補語と検索クエリの AND 検索のヒット件数（件数 B)」の関係を考慮したランキングを行った．件数 B は，候補語と検索クエリの AND 検索を行っているため，件数 A よりも件数 B は小さい値となる．また，候補語と検索クエリの AND 検索で得られた結果は候補語の検索で得られた結果と包含関係にある．各候補語のスコアは，候補語の検索結果の中でどのくらい候補語と検索クエリの結果を網羅しているかを計算した値（件数 B/件数 A）が付けられる．このスコアが高い順にランキングされる．

3 つ目では，候補語の idf を計算しその値でランキングを行った．idf は，式 1 から求めた．dt(t) は単語 t の出現する文書数を表し，N は比較文書数である．ここで，dt(t) は，Yahoo!検索 API を利用した時の検索クエリ t に対する検索結果ヒット件数である．また，比較文書数 N は 548 億に設定する．この数値は，Yahoo の検索エンジンを用い，検索クエリとして「www」や「http」など多くの Web ページがヒットするような検索クエリで検索した時のヒット件数である．検索した時のヒット件数よりも多い値となるように仮に設定した値であり，Web ページ全体のページ数を正確に反映したものではない．このようにして，式 1 から得られるスコアの高い順にランキングを行った．

4 つ目は，Cody らの質問応答システムのランキング [5] を参考にした．まず，答え候補の中で，文字数が等しいものを一つのクラスタとする．次に、そのクラスタに属する単語の数でクラスタをランキングする．最後に，各クラスタのラベルとなる単語を決定する．その単語は，答え候補における各単語の出現数を計算し，その値が一番高かったものを各クラスタのラベルとした．

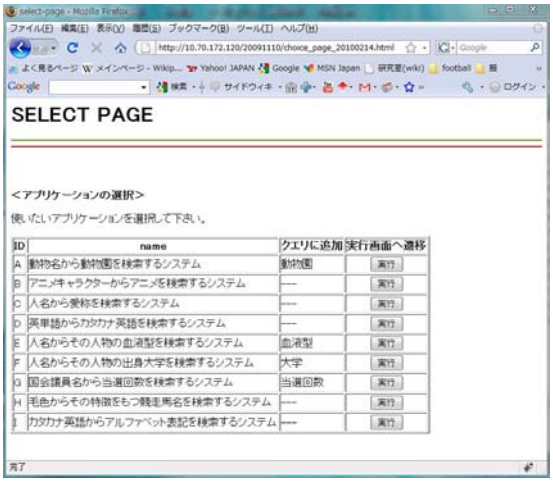

図 4　生成したアプリケーションの一覧画面

$$idf = log\frac{N}{dt(t)} \tag{1}$$

3.4　システムの構成と実行例

本システムは，アプリケーション作成部，アプリケーション選択部，およびアプリケーション実行部の 3 つの部分から構成されている．アプリケーション作成部では，アプリケーションを作成するユーザがシステムにいくつかの答えを教え，アプリケーションを作成する．アプリケーション選択部では，アプリケーションを使うユーザが自分の使いたいアプリケーションを選択する．アプリケーション実行部では，アプリケーションを使うユーザが検索語を入力し，表示された結果を閲覧する．

図 2 は，アプリケーション作成部の入力画面の一例である．上から順番に，第 3.2 節で説明した正例の入力，語の切り分け方法の選択，スコアリング関数の選択，学習アルゴリズムの選択を行い，アプリケーションを作成する．図 4 は，アプリケーション選択画面の一例である．アプリケーションを使うユーザが使いたいアプリケーション名横のチェックボックスにチェックを入れて選択ボタンを押下する．図 3 は，アプリケーション実行部の画面の一例である．上部の検索ボックスに検索語を入力し，検索ボックス隣の output ボタンを押下すると結果が下部に出力される．

アプリケーションを作成するユーザが作成したアプリケーションの公開方法の一つとして、ブログを扱った方法を紹介する。作成されたアプリケーション画面に iframe 用のリンクが用意してある。そのリンクをコピーする（図 5)。次に、自分が使用しているブログの本文にコピーしたものを張り付け保存をする（図 6)。保存が終わり結果が反映されると、実際にブログを見た人が作成したアプリケーションを使用できる状態となる (図 7)。

4.　評　　価

4.1　実験条件と評価方法

本手法による評価実験として，表 3 に示した 8 個のアプリケーションを実際に作成した．また，アプリケーション作成ユーザが用意した正解なのか正解ではないのかという二値クラス

図 5　作成したアプリケーションの使い方：ステップ 1

図 6　作成したアプリケーションの使い方：ステップ 2

図 7　ブログでのアプリケーション利用画面

表 2　confusion matrix の例

	class(predicted)	not-class(predicted)
class(true)	34	0
not-class(true)	2	32

分類問題 [10] として扱い，再現率（recall）と精度（precision）をそれぞれ式 2，式 3 のように定義し，評価を行った．なお，M(0,0) は表 2 の matrix の class(true) かつ class(predicted)，すなわち，そのクラスに属すると予測され，実際にそのクラスに属した場合の事例数を意味する．同様に M(0,1)，M(1,0) もそれぞれ class(true) かつ not-class(predicted)，not-class(true) かつ class(predicted) を意味する．正例の準備は被験者となる大学生 8 人が行った．

アプリケーション A は，動物名を被験者に提示させた．動物名で Yahoo!検索しても動物園名をあまり得ることができなかったことから検索語に動物園を足した．「被験者の提示した動物名」＋“ 動物園 ”で Yahoo!検索を行い，上位 50 件の中でその動物が見られると推測される動物園名をサマリから抜き出し，動物名＋文字列“ 動物園 ”と動物園名（複数）の正例を 13 個用意した．語の切り分け方法は，形態素解析の結果をそのまま用いて，名詞が連続する場合のみ連続する名詞を一つの名詞として扱うようにした．スコアリング関数は，関数 1，重要な単語（動物園）は頻出することが予想されることから関数 2，そして，「○○動物園」という文字列が答えである可能性が高いと考えられるため関数 9 を用いた．なお，関数 9 の対象の文字列は「動物園」とした．

表 3　構築したアプリケーション一覧

システム *ID*	作成したアプリケーション	正例数
A	動物名から動物園を検索	13
B	アニメキャラクターからアニメを検索	18
C	人名から俗称を検索	32
D	英単語からカタカナ英語を検索	50
E	人名からその人物の血液型を検索	25
F	人名からその人物の出身大学を検索	25
G	国会議員名から当選回数を検索	17
H	毛色からその特徴をもつ競争馬名を検索	8

アプリケーション B は，アニメキャラクターとそのキャラクターが登場するアニメを被験者に提示させ，アニメキャラクター名と登場するアニメの正例として 18 個用意した．語の切り分け方法はアプリケーション A と同様である．スコアリング関数は，関数 1，キャラクターが関係するアニメ名が文章内に頻出することが予想されるため関数 2 を用いた．

アプリケーション C は，芸能人の名前とその芸能人の愛称を被験者に提示させ，芸能人の名前と愛称の正例として 32 個用意した．語の切り分け方法はアプリケーション A と同様である．スコアリング関数は関数 1，芸能人名で検索した場合に文章内に愛称が頻出すると予想されるため関数 2，また「愛称こと芸能人名」というように愛称と芸能人名の単語間の文字数がある程度近いことが予想されるため関数 10 を用いた．

以下，同様に，被験者の判断に基づきアプリケーション D からアプリケーション H を本自動構築機構を用いて作成した．

本提案手法では，データマイニングツール Weka（Waikato Environment for Knowledge Analysis）[11] を用いて分類学習を行っている．学習アルゴリズムは，C4.5，NaiveBayes，SVM（カーネルは linear を使用）の 3 つを使用し，実装は Weka および LibSVM [12] [13] を用いた．評価方法は 10 fold cross-validation とした．

$$recall = \frac{|M(0,0)|}{|M(0,0) + M(0,1)|} \tag{2}$$

$$precision = \frac{|M(0,0)|}{|M(0,0) + M(1,0)|} \tag{3}$$

4.2　結果と考察

図 8，図 9，図 10 には，各アプリケーションの精度（precision），再現率（recall），およびその調和平均（F 値）を示す．いずれのアプリケーションでも F 値として 0.65 以上の値が得

られた．

特に，アプリケーション D，F，および G は F 値が高かった．その要因としては，アプリケーション F，G は，それぞれの答えが“ ○○大学 ”や“ ○回 ”というように答えにある程度の規則性があるため良い結果が得られたと考えられる．また，形態素解析においても，それぞれ「漢字」＋“ 大学 ”,「数字」＋“ 回 ”という単純な組み合わせなので，答えの文字列が意図せぬところで複数の形態素に分かれてしまうことが少ない可能性があると考えられる．

このように，答えにある程度の規則性を持つ文字の並び方がある場合には，語の切り分けは指定せず形態素解析の結果をそのまま用いても良い結果が得られると予想される．

アプリケーション D では，答えはカタカナのみの文字列であり，また英単語の文字数と答えの文字数にある程度の関係があることが予想される．さらに，英単語の先頭文字と答えの先頭文字にもある程度の関係がある．後者に対して簡単に試作したスコアリング関数 11 をアプリケーション D の生成に利用したことが良い結果を得られた要因であると予想される．このスコアリング関数は，他の関数と異なり検索語がアルファベットのときにのみ有用である．さらに，1 つのアルファベットに対して子音なら 5 つ（アルファベットが’t’ ならば,“ タ，チ，ツ，テ，ト ”），母音なら 1 つ（アルファベット’a’ ならば，’ア ’）とすることで大まかな絞り込みが可能である．

特殊な（あるアプリケーションだけに有用な）関数を用いれば，さらに精度などを向上できると考えられるが，そのような特殊なスコアリング関数を網羅するには限界がある．

アプリケーション F，G のように答えとする文字列に特定の規則性がなくても，有効なスコアリング関数を作成し利用することで良い結果を得られることがあることがわかった．

アプリケーション B，C，および E は，F 値が他に比べ低かった．アプリケーション B，C は，アプリケーション F，G と比べて答えに規則性がないことが F 値が低かった原因であると考えられる．さらに，アプリケーション D のような特殊なスコアリング関数を利用していないことも原因の一つであると予想される．

アプリケーション E は，アプリケーション D 同様に答えが「アルファベット」というように答えの文字の種類が分かっている．さらに，おおざっぱな答えの種類は“ A，B，O，AB ”の 4 種類である．しかし，例えば答え“ a ”ならば“ abstract,star ”のように，それらの答えが他のアルファベットの文字列の一部分に含まれていることが多いため，precision が低かったと考えられる．この場合には，形態素解析において「アルファベット」と“ 型 ”が分かれないように指定することで，アプリケーション F，G のように良い結果が得られると考えられる．

答えの種類数よりも答えにある程度の規則性があることが本システムにおいて高い精度が期待できる可能性があることが分かった．

また，アプリケーション B において，いずれのアルゴリズムでも F 値に大きな差はないが，C4.5 を用いた場合には精度が高く，NaiveBayes や LibSVM を用いた場合には再現率が高くなっていた．このことから，精度と再現率においてユーザがより重きを置くほうを得られるように学習アルゴリズムによって使い分けることが考えられる．

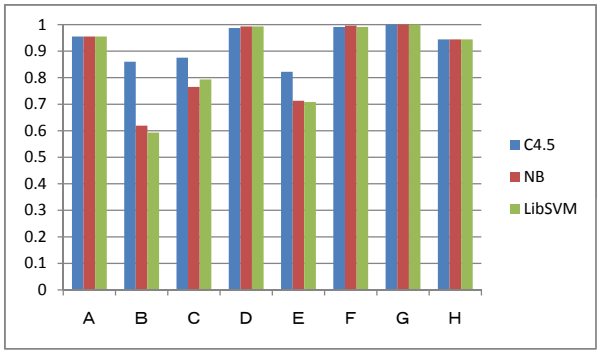

図 8　各アプリケーションの解答精度

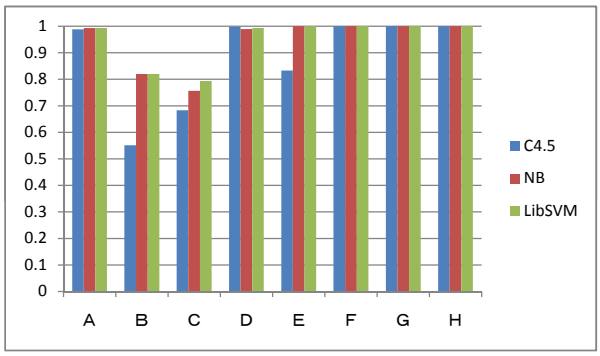

図 9　各アプリケーションの解答再現率

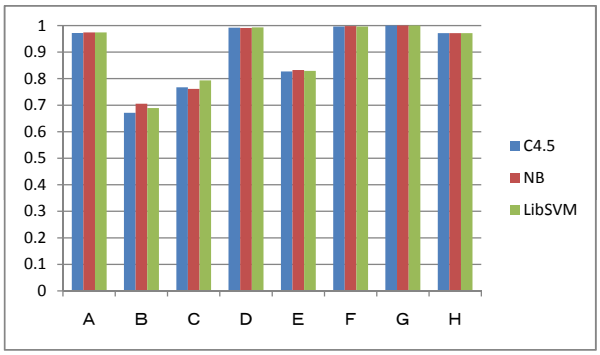

図 10　各アプリケーションの解答性能（F 値）

5. 議　　論

機械学習時に用いる属性の選択方法において，既存の属性選択手法の適用・改良をすることで属性の性能ミスによる性能低下を抑止できる可能性がある．属性の選択が，生成するアプリケーションの性能に与える影響について，アプリケーション作成時に用意する正例データの数，スコアリング関数の選択数，および，どのスコアリング関数を選択すればよいか，の 3 つの点から検討を行った．題材として，英単語を入力するとその英単語のカタカナ表記を提示するカタカナ英語検索アプリケーションを扱った．用意した正例データは 100 件で，スコアリング関数の数は 11 種類を用意した．

1 つ目の検討事項「アプリケーションを作成するユーザは正

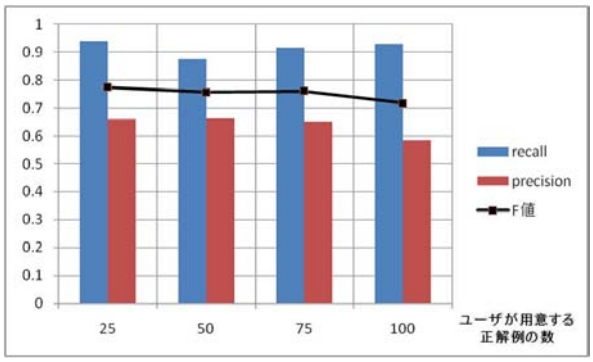

図 11　用意した正答数と性能の関係

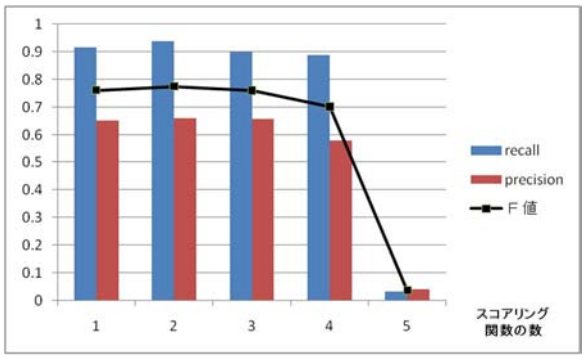

図 12　使用スコアリング関数の数と性能の関係

解データをどのくらい用意すれば性能が低下しないだろうか」については，ユーザの用意するデータを 25 件，50 件，75 件，100 件と変化させていき，一番高い再現率が 25 件，一番高い精度が 50 件となり，F 値は，25 件が一番高い結果となった（図 11）．平均して 0.7 以上の F 値が得られ，25 件程度の少量の正例であっても十分な性能が得られていた．

2 つ目の検討事項として「アプリケーションを作成するユーザはスコアリング関数をいくつ選択すればよいのだろうか」ということを検討した．スコアリング関数の選択数を 1 から 5 まで増やしていき性能を確認した（図 12）．スコアリング関数を 5 個選択したときは結果がきわめて悪く，通常は 2 か 3 程度のスコアリング関数を選択すれば十分な性能が発揮できている．1 つのみスコアリング関数を用いるときもある程度良好な結果が得られているが，結果の一般性に関しては，スコアリング関数と対象とする問題との相性など，様々な要因を考慮する必要がある．

3 つ目の検討事項として「アプリケーションを作成するユーザはどのスコアリング関数を選択すればよいのだろうか」ということを検討した．検討したスコアリング関数は，11 個のうちスコアがすべて 0 とならなかった 1，2，3，10，11 で検討を行った（図 13）．F 値の平均が高かったのが，2 番と 11 番であり，2 つ目の検討事項の結果とも合わせて考えるとこの 2 つを選択するのがよいと考えられる．

以上より，アプリケーションを作成するユーザに対してスコアリング関数の推薦が行えれば有益であろうと考えられる．ユーザの用意した正解例をもとに，スコアリング関数に対して，それ単体を選んだ場合に F 値の平均が高い順にリストを作成し，上位 3 つをユーザに推薦をするなどの方法が考えられる．

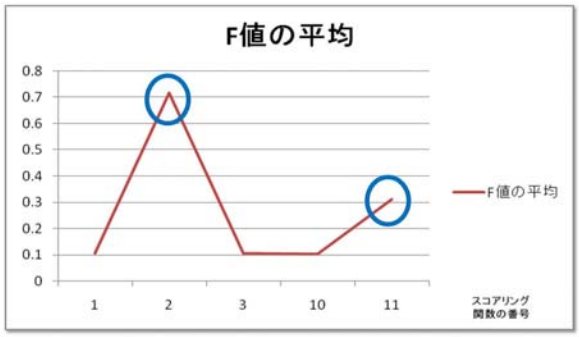

図 13　各スコアリング関数と性能（F 値）との関係

6. 関 連 研 究

Baykan らは，URL のみが与えられた状況からその言語を判断する手法を提案した [14]．Baykan らの手法では，URL に対して， trigram や辞書を利用して特徴を抽出し機械学習によって言語を判断している．本研究のアプローチは，言葉の判断に特化したものではなく様々な質問応答型検索に適用できる手法を提案している．

検索エンジンのサマリの統計的な情報などを用いた多言語用例検索システム Kiwi システム [15] では，Web 上で実際に使用されている用例を抽出してユーザに提示し，Web 上の大量の情報の統計的側面から適切な用例・表現を判断しやすくしているが，用例の提示に特化しており，本研究でめざすような質問応答型検索システムの自動構築手法は示されない．

Web 検索を利用して与えられた語と特定の関係にある語を発見する手法がある [16]．例えば,「松井秀喜」に対して「ゴジラ」といった俗称を求めるもので,「ゴジラ」という呼称は「こと松井秀喜」というフレーズの直前に現れるという発見手法の一般化を提案している．さらに，Web 検索結果（サマリ）という少ない量のテキストからでも比較的精度よく高速に語の発見を行っている．本研究では，サマリのみを対象とすることにより高速化を図ることも可能であり，また，対象の文書を増やしたい場合にはページ本文を扱うことも可能であり，ユーザが選択をすることができる．両方向構文パターンという距離に限らず，与えられた語と特定の関係に語を発見する手法を提案した．

Marius らは，質問応答型システムにおいて答えのタイプの分類（推定）に着目した [6]．WordNet（概念辞書）を利用してユーザの質問文から答えのタイプの予測を行い，その予測したタイプの単語を含む文章を得られるように検索キーワードの追加・削除などを調整することで解答精度を高める提案をした．本研究では，ユーザは質問文全体ではなくその部分のみの穴埋めとすることにより質問文の解析・答えのタイプの予測を簡略化した．

佐々木らは，機械学習を用いて日本語 QA システムの主要なコンポーネントをそれぞれ学習データから構築することにより，QA システム全体を構築した [7]．質問文およびその正解例か

ら，質問文に対する解答法を学習する QA システム全体の構成法について述べていた．その中で，質問解析と解答選択の 2 つのモジュールを分類問題として捉え，各分類器を質問文とその正解例から学習する方法，および学習した分類器の利用方法を明らかにした．本研究では，入力させる対象を質問文全体ではなくその部分のみの穴埋めとすることにより，質問文の解析を簡略化し，機械学習の適用対象を解答選択モジュールに限定することで，入力文解析精度と回答品質の向上を狙っている点で異なる．

質問に解答するための情報源として，検索エンジンから返されるページのサマリのみを利用した質問応答型検索システムの研究が行われている [17]．このシステムでは，質問が与えれらた場合に質問の答えが記述されていそうな単語を加えたいくつかの string を生成する．検索結果のページのサマリは，クエリ（質問文）を含んでおり，おおよそそのクエリの周りにはいくつかの（質問の答えである可能性の）単語がある．あるページの全文ではなくサマリという制約のもとでも与えられた質問からクエリ生成をする際にクエリの拡張及び複数のクエリの用意をすることで質問応答型システムを実現している．

既存の質問応答型検索システムは，ある質問に対してその質問に対する明確な答えを含む文章（ページ）を探すということを行っている [18]．しかしながら，ある質問に対する答えが 1 つの文章（ページ）だけでは推定できないことがある．このような場合に既存の質問応答型検索システムでは対応が困難な場合がある．この問題解決のために，Stefan らは「the HOLMES system」というテキスト推定を行うシステムを実装した．そのシステムでは，WordNet などの知識モデル・推定ルールを用いて Web という巨大なテキストコーパスから答えを提示するシステムである．本研究では，Web 上の情報に対してヒューリスティックスに，質問に対する答えを提示するシステムを提案した．

質問応答システムへの質問の中でも，事実に基づいた質問に対して間違った答えを与えてしまうという問題がある．そのような場合に矛盾検知を行うということが考えられる．例えば，質問としてある人物の出身地を聞いたときに，地理的に異なる 2 つの場所が提示された場合には矛盾が発生する．Alan らは，WordNet から包含関係を考慮した矛盾検知システム「AuCONTRAIRE CD system」を実装した [19]．本研究では，ウェブからの情報の統計的な側面を重視し，答えを提示する仕組みとなっているため，提示される答えの間の矛盾については考慮していない．

7. 今後の課題

今後の課題としては，検索アプリケーション自体を JavaScript 化して自由にページ内に張り付けることができるようにすることで，検索エンジン以外に特定のサーバを必要とせずに，負荷を分散しながら実行できるようにすることが挙げられる．

ユーザの質問に対する解答提示方法も今後の課題である．本自動構築機構で生成したアプリケーションにおいて，正答が一つのみであり得られた答えが複数ある場合のランキングについて実装を行った．その性能評価や，正答が複数ある場合の順位づけ，及び，その効果的な提示方法などを検討する必要がある．

また，外部で作成した Web サービス等をスコアリング関数として利用可能にできれば，アプリケーションを作成する際に，本システムで用意したスコアリング関数の中に適切なスコアリング関数が存在しない場合にも柔軟な対応が可能になると考えられる．

謝辞 本研究の一部は科研費基盤 B(19300026) の助成を受けたものである．

文　献

[1] 松尾豊： 世界へのインタフェースとしての検索エンジン，電子情報通信学会誌,Vol.90,No.10,pp.884-888,2007.
[2] Yahoo!知恵袋，http://chiebukuro.yahoo.co.jp/.
[3] 教えて！goo，http://oshiete.goo.ne.jp/.
[4] 栗山和子，神門典子：Q & A サイトにおける質問と回答の分析，第 148 回 データベースシステム 第 95 回 情報学基礎 合同研究発表会,2009.
[5] Cody C. T. Kwok, Oren Etzioni, and Daniel S. Weld.：Scaling question answering to the Web，In Tenth International World Wide Web Conference (WWW-10), Hong Kong, May 2001.
[6] M. A. Pasca and S. M. Harabagiu：High performance question/answering.，In Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR ’2001), 366-374.
[7] 佐々木裕, 磯崎秀樹, 鈴木潤, 国領弘治, 平尾努, 賀沢秀人, 前田英作：SVM を用いた学習型質問応答システム SAIQA-II，情報処理学会論文誌, Vol.45,No.2 , pp.635-646, 2004.
[8] 島崎祐輔，横山昌平，福田直樹，石川博：特定用途向け簡易 Web 検索システムの自動構築に向けて， IEICE 総合大会，D-8-1, 2009.
[9] Yahoo! デベロッパーネットワーク，http://developer.yahoo.co.jp/
[10] Soumen Chakrabarti： Mining the Web:Discovering Knowledge from Hypertext Data，Morgan Kaufmann,2002.
[11] Ian H. Witten and Eibe Frank.： Data Mining: Practical machine learning tools and techniques.，Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005.
[12] Rong-En Fan, Pai-Hsuen Chen, and Chih-Jen Lin.：Working set selection using second order information for training support vector machines.，J. Mach. Learn. Res., Vol. 6, pp. 1889-1918, 2005.
[13] Chih-Chung Chang and Chih-Jen Lin.：LIBSVM: a library for support vector machines, 2001.， Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.
[14] E.Baykan, M.Henzinger and I.Weber.：Web Page Language Identification Based on URLs.，vldb2008,pages 167-187.
[15] Kumiko Tanaka-Ishii,Hiroshi Nakagawa： A Multilingual Usage Consultation Tool based on Internet Searching :More than search engine, Less than QA， The 14th Internatinal Wordl Wide Web Conference(WWW2005) pp.363-371.2005, May,Chiba,Japan.
[16] 大島裕明, 田中克己：両方向構文パターンを用いた Web 検索エンジンからの高速関連語発見手法， 情報処理学会研究報告, 2008, 88, pp. 37-42(2008).
[17] Dumais, Susan, Michele Banko, Eric Brill,Jimmy Lin, and Andrew Ng：Web question answering: Is more always better?， In Proceedings of the 25th ACM SIGIR, pages 291-298, Tampere, Finland,2002.
[18] Stefan Schoenmackers,Oren Etzioni,and Daniel S.Weld：Scaling Textual Inference to the Web， In Procs. of EMNLP,2008.
[19] Alan Ritter,Doug Downey,Stephen Soderland and Oren Etzioni：It’s Contradiction – No,It’s Not: A Case Study using Functional Relations， In EMNLP,2008.