DEIM2021 F31-3

知識ベースを活用した探索的な文書検索

阿曽　太郎† 　天笠　俊之†† 　北川　博之††

† 筑波大学システム情報工学研究科　〒 305–8573 茨城県つくば市天王台 1-1-1
†† 筑波大学計算科学研究センター　〒 305–8573 茨城県つくば市天王台 1-1-1
E-mail: †aso@kde.cs.tsukuba.ac.jp, ††{amagasa,kitagawa}@cs.tsukuba.ac.jp

あらまし　企業やコミュニティは，様々な情報を文書形式で蓄積している．そうした文書集合の内容を調べることで，その組織に関する知識や状況を知ろうとすることは，重要な作業となっている．しかし，その組織に詳しくないユーザは，問合せに必要となる文書集合に関係する知識を十分に持たないため，効率的に問合せできず，結果としてその作業には多くの労力がかかる．こうした問題に対し，我々は文書集合に関する知識を抽出し，RDF 形式の知識ベースを構築することで，関係する知識を検索できるようにする．また，非専門家であるカジュアルなユーザにとって知識ベースの構造や語彙を理解して SPARQL 問合せを行うことは難しいため，Query by Example ベースの検索インターフェースによって，関係する知識を利用した検索を簡単化することを提案する．

キーワード　知識ベース，RDF，探索的検索，文書検索

1 はじめに

企業やコミュニティはさまざまな情報を文書形式で蓄積している．蓄積された文書の内容を調べることで，その組織や関係する分野に関する情報を得ようとすることはよくあり，重要な作業となっている．例えば，(1) あるエンティティに関する文書を調べたい，(2) あるエンティティの関連事項に関する文書を調べたい，(3) あるエンティティが特定の文脈で言及された文書を調べたい，などのニーズがある．

対象の文書集合やその内容について詳しくない人にとって，そうした調べごとには多くの労力がかかる．例えば，一般的な文書検索システムを利用して調べごとを行う場合，ユーザはキーワード検索機能や文書ファイルに付随するメタデータなどを利用したファセット検索機能を利用して，タスクに関係しそうな文書を取得しようとする．しかし，検索対象の文書集合やそれらの対象分野に詳しくない人が，検索意図を具体的に明示することは難しい．また，文書ファイルのメタデータによるファセットは，文書の内容に関する情報が十分ではない場合が多い．結果として，ユーザは効率的に問合せすることができず，文書の内容を確認しながら必要な文書を判断するため，多くの労力を費やしてしまう．

この問題を解決するには，ユーザが文書集合に関するドメイン知識を参照・利用しながら，探索的に検索ができるようになればよい．本稿では，その具体的な方法として，文書の外形的なメタデータだけではなく，文書のコンテンツに関する情報も含んだ包括的な知識ベースを構築し，構築した知識ベースを利用した文書検索手法を提案する．

知識ベースとは，世の中の事物に関する知識を構造化して蓄積したデータベースのことである．知識ベースの記述には，Resource Description Framework（RDF）が用いられる．RDF とは，ウェブ上で情報を表現するための枠組みで，RDF においては，世の中の事物や概念，属性といったあらゆるものがリソースとして扱われる．リソースは，Uniform Resource Identifier（URI）で識別され，あるリソースについての 1 つの情報は，主語（Subject），述語（Predicate），目的語（Object）から構成される 3 つ組（トリプル）のグラフ構造で記述される (図 1)．主語は情報を記述される対象のリソースを示し，述語は主語に関する情報のプロパティを定義する．そして，目的語は述語の対象である．主語と述語は URI で記述し，目的語は URI もしくは数値や文字列などのリテラルで記述する．RDF は，さまざまなエンティティやその属性，エンティティ間の関係性などを表現しやすく，規格化されているため機械処理がしやすい．したがって，多種多様な情報を格納・利用するのに適している．我々は，文書の作成日や作成者，タイトルなどの外形的な情報と，文書テキスト中で言及されるエンティティやエンティティの言及内容といったコンテンツに関する情報を合わせた，包括的な RDF 形式の知識ベースを構築することで，文書集合に関するドメイン知識を参照・利用した高度な検索ができるようにする．

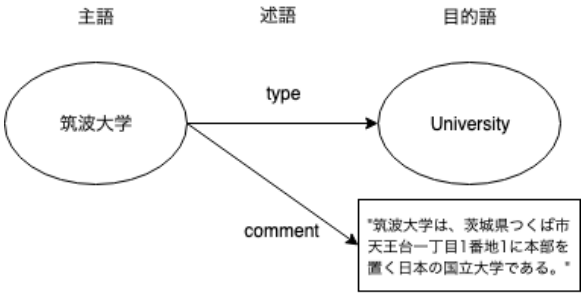

図 1　RDF の例

RDF 形式の知識ベースを利用した文書検索を行うには，問合せ言語 SPARQL を利用できる．SPARQL の文法に従って，必要なトリプルパターンを集めたグラフパターンを組み立てる

ことで，欲しい文書を取り出すことができる．しかし，一般的なユーザにとって，SPARQL 検索を行うハードルは高い．なぜなら，まず，SPARQL の文法を理解し習得する必要がある．そして，検索対象の知識ベースに格納されている事物や使用されている属性の URI について理解する必要がある．知識ベースは様々な種類の事物や属性が存在する複雑な構造になっているため，特に後者の理解は専門家であっても簡単ではない．

この問題に対して，我々は，Query by Example（QBE）[14] を基に着想した探索的な検索手法を提案する．QBE とは，関係データベース向けの視覚的な問合せ言語である．関係は表で表現され，ユーザは表の行や列を選択し，条件などを記載して，問合せを実行する．SQL の文法や表現を知らない場合でも，感覚的に問合せ操作を実行することができる手法である．提案手法では，ユーザのニーズに応えるために必要であると思われる文書集合に関する知識を事前に設定し，その知識を表形式で提示することで，知識ベース中の知識を参照できるようにする．そして，表中の行を選択することでその知識に関する文書検索やその知識に関係するさらなる知識の取得を可能にする．

本研究の貢献は次のとおりである．

（1） 文書集合に関する包括的な知識を格納する知識ベースを構築し，その知識を参照・利用した高度な検索を可能にした．

（2） Query by Example ベースの探索的な検索システムを実装し，非専門家であるカジュアルなユーザが検索できるようにした．

（3） 提案手法を実世界のデータセットに適用し，特定の検索ニーズにおいて，一般的な全文検索エンジンを利用した文書検索より，提案手法が Precision で優れていることを確認した．

2 前提知識

本節では，前提知識として，RDF の記述と SPARQL について述べる．

2.1 RDF の記述

RDF はあらゆる事物をリソースとして扱い，リソースの性質やリソース間の関係を記述することができる．記述者が自由に URI を設定して記述することもできるが，一般的にはその性質や関係ごとのオントロジー（概念体系）で定義される語彙を利用することが推奨されている．例えば，人間や組織の分野とデータベースの分野の概念体系は異なる．そのため，人間や組織の関係を記述する語彙を提供するオントロジーとしては FOAF[1]が，データベース間の関係を記述する語彙を提供するオントロジーとしては VoID[2]が整備されている．各オントロジーにはそれぞれの語彙を規定するための名前空間が存在する．知識ベースは，リソースやリソース間の関係の種類に応じて，オントロジーや語彙を使い分けて記述することで構築される．

2.2 SPARQL

RDF 形式の知識ベースに対する問合せ言語として，SPARQL がある．SPARQL 処理系は、クエリからグラフパターンを構築し，そのパターンに一致するリソースを RDF 集合の中から探し出す．SPARQL クエリにおいて，複数の名前空間の URI が記述される場合は，PREFIX 句で名前空間を宣言しておくことができる．変数は “?” 文字から始めることで宣言できる．この変数を使って見つけ出すグラフパターンを WHERE 節に記述し，結果に表示する変数を SELECT 句に記述する．WHERE 節ではそれぞれの変数について FILTER 句を用いて条件を記述することができる．グループ化演算子（GROUP BY）を利用した集約演算や順序づけ演算子（ORDER BY）も利用できる．また，OPTIONAL 句によって指定されたグラフパターンは補助的なグラフパターン として処理される．つまり，そのグラフパターン が存在する場合は，問合せに考慮されるが，存在しない場合は無視される．UNION 句を利用して複数種類のグラフパターン に対する問合せ結果を結合することもできる．

3 関連研究

3.1 知識ベースの構築

さまざまな目的において，知識ベースの構築が提案されている．Yen ら [12] は，ライフログの記録を目的として，Twitter 上で共有されたテキストデータからライフイベントを抽出し，個人の知識ベースを構築することを提案している．Oramas ら [7] は，Web 上の音楽関係の情報から自然言語処理の技術を使って，音楽分野の知識ベースを構築し，音楽の推薦における有用性を評価している．Tchechmedjiev ら [9] は，Web 上で議論される様々な言説のファクトチェックを支援することを目的として，ClaimsKG という知識ベースを提案している．ClaimsKG はファクトチェックを行う複数のウェブサイトから抽出した情報を基に構築されている．また，本研究と類似する研究としては，Chirita ら [2] が，デスクトップサーチへの利用を目的として，ファイルのメタデータやディレクトリ構造，ブラウザのキャッシュ情報などを RDF データ化することを提案している．しかし，本研究では，ファイルのメタデータに加えて，ファイルのコンテンツ情報に関しても RDF データ化し，検索に利用することを提案している．一方で，RDF データ化については，ClaimsKG のデータモデルや利用されている語彙を参考にしている．

3.2 知識ベースを利用した探索的な検索

さまざまな目的において，知識ベースを利用した探索的な検索が提案されている．Waitelonis らは [11]，動画データの探索的意味検索を容易にするために，Linked Open Data と呼ばれる複数の知識ベースの集合体の利用方法を提案している．Hahn ら [5] は，Wikipedia の探索的な検索を目的として，Wikipedia の情報を基に構築された DBpedia を利用したファセット検索の手法を提案している．また，Yogev ら [13] は，知識ベースなどの Entity-Relationship データに対して，直感的なクエリ言語とファセット検索とグラフナビゲーションを組み合わせた探索的な検索手法を提案し，社会医学分野での適用例を発表している．本研究と類似した研究としては，Bhagdev ら [1] が，オ

1 : http://xmlns.com/foaf/spec/

2 : https://www.w3.org/TR/void/

ントロジーベースの検索とキーワードベースの検索を柔軟に組み合わせることで，文書検索と知識検索の両方を支援するハイブリッド検索を提案している．ハイブリッド検索では，セマンティック検索手法の限界の一つである文書内容のセマンティックカバレッジの欠如への対応として，情報抽出技術を利用している．しかし，抽出された情報を基にした具体的な知識ベースの構築やデータモデルについては言及がない．また，探索的な検索は目的ではなく，検索システムを利用するユーザは文書集合のコンテンツに関して十分な知識を持っていることが前提とされている．よって，本研究とは，知識ベースの構築方法の議論がない部分と問題設定が異なっている．

4 提案手法

我々は，図 2 のシステム概要に基づいて，文書集合に関する知識を利用した探索的な文書検索システムを提案する．本システムの目的は，知識ベースの情報を利用した高度な問合せによって，ユーザの興味に応じた文書を検索できるようにすることである．ユーザの操作とシステムで実行される処理は次のとおりである．

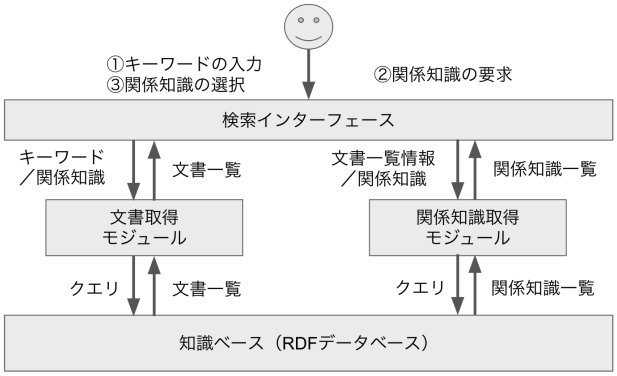

図 2　システム概要

（1） ユーザはキーワード検索を実行する．システムは文書取得モジュールにて，入力されたキーワードを含む SPARQL クエリを生成し，知識ベースに対して問合せする．問合せ結果が検索インターフェースに返却され，表形式に整形されて表示される．

（2） ユーザは結果の文書一覧に関係する情報を要求できる．要求されると，文書一覧情報が関係知識取得モジュールに入力される．関係知識取得モジュールは，入力された文書一覧に関する，事前に定義された知識を問合せる SPARQL クエリを生成し，知識ベースに対して問合せする．取得された知識は表形式に整形されて表示される．

（3） ユーザは関係知識を選択し，その知識に関係する文書を取得することができる．また，関係知識に対してさらなる関係知識を要求することができる．文書取得のクエリは，(1) の SPARQL クエリに対して，選択された関係知識のトリプルパターンを追加することで，生成される．関係知識のさらなる取得も，事前に定義された SPARQL 問合せに選択された知識に関するトリプルパターンを追加することで，生成される．結果の返却は (1)，(2) と同様である．

本手法のポイントは 2 つある．1 つは，文書に関する多種多様な情報を RDF 形式のトリプルとして抽出し，知識ベースに格納することで，単純なテキスト検索では利用しにくい情報を利用した高度な検索が可能になることである．次に，あらかじめ検索に有用であると思われる知識ベース中の知識を設定し表形式で提示することで，グラフ中の複雑な関係を簡潔に示し，ユーザが知識ベースのデータ構造や語彙を知らなくても，関係する知識を利用した高度な検索を可能にすることである．

以降，構築する知識ベース（RDF データベース）の概要とその構築方法について説明し，その後，検索システムの概要とその処理内容について説明する．

4.1 知識ベース

知識ベース構築の目的は，文書集合に関するさまざまな知識を取得・利用した高度な検索を可能にすることである．そのため，文書の外形的な情報，内容（コンテンツ）に関する情報，コンテンツに関する一般的な情報を統合した包括的な知識ベースを構築する．一般的には，文書集合に関する知識ベースを作る場合，その内容に適したオントロジーを構築し，それに従って文書集合の情報を RDF トリプルとして記述する．しかし，オントロジーの構築にはその分野の専門家が必要となり，コストがかかる．また，対象となる分野が異なれば，分野ごとにオントロジーを構築する必要性も生じ，負担が大きい．そのため，本稿ではオントロジーの構築にコストをかけずに，文書集合に関する情報を統合する知識ベースのモデルを提案する．図 3 は提案モデルを具体的なデータを使って示した図である．以降で，それぞれの情報源と構築方法について説明する．なお，このデータモデルでは，具体的な文書例として，メール文書を想定して説明する．

4.1.1 文書ファイルのメタデータ

文書ファイルのメタデータを抽出し RDF トリプルにすることで，文書間の関係性などを利用できるようにする．文書ファイルのメタデータとは，ファイルの作成者や作成日，ファイル名などファイル単位で付属する情報のことを指す．メール文書の場合，メールのヘッダ情報が該当する．具体的には，メール識別子，メールタイトル，送信日，送信者，受信者，送信者のメールアドレス，受信者のメールアドレス，返信先メールの情報などがある．

具体例が，図 3 の破線で囲まれた **1. 文書ファイルのメタデータ**部分である．メールエンティティは，schema:EmailMessage のクラスに属し，メール識別子を変換した URI でリソースとして表現する．そして，ヘッダ情報の各属性を Schema.org[3]や FOAF などのオントロジーで定義される語彙を使って表現する．それらのオブジェクトがエンティティの場合は URI によるリソースとして記述し，日付やラベルなどの文字列の場合はリテラルとして記述する．また，RDF の基本語彙の 1 つである rdf:type を使って，エンティティのクラス情報を定義する．

3 : https://schema.org/

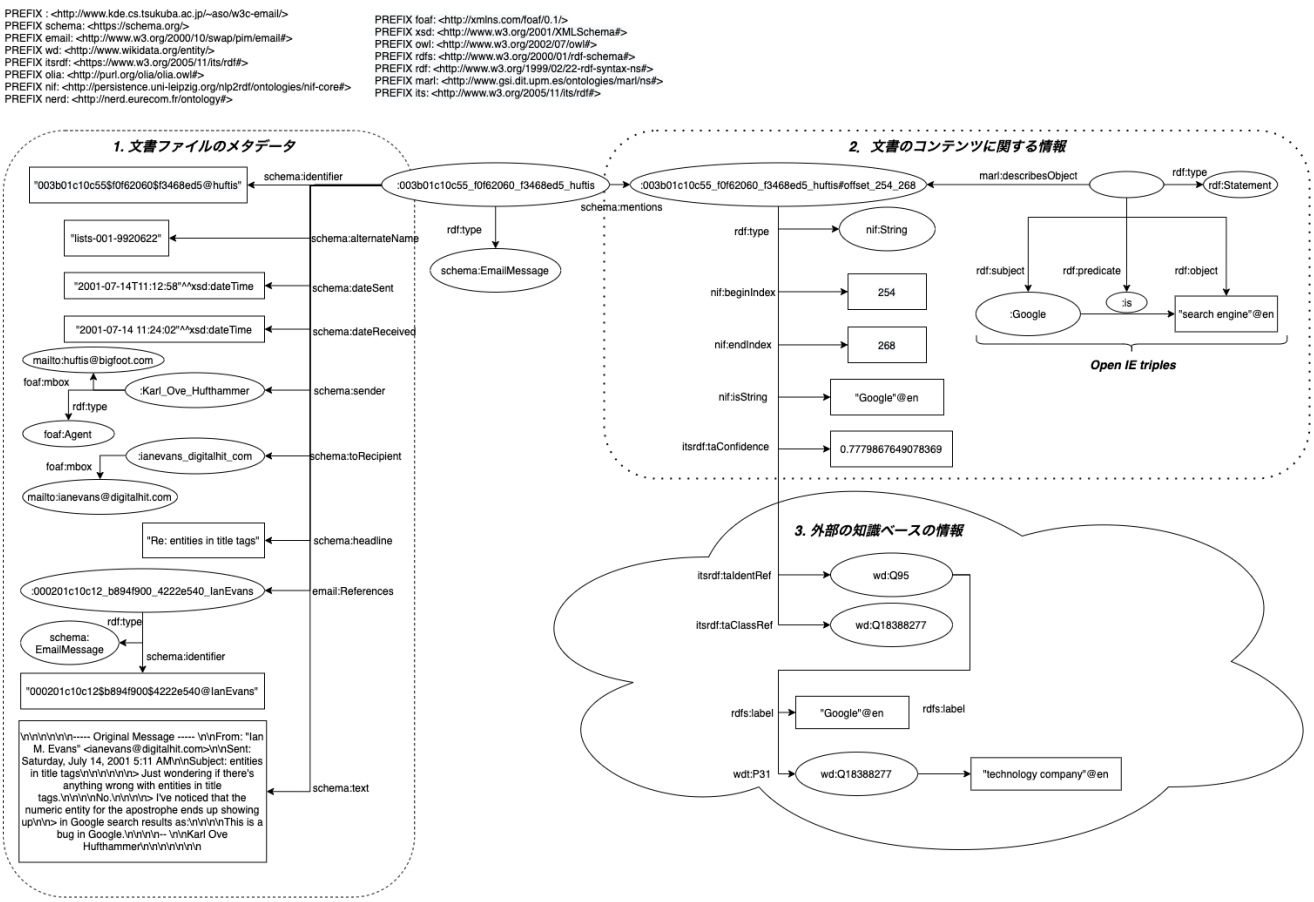

図 3　知識ベースのデータモデルを具体化した図

4.1.2　文書のコンテンツに関する情報

文書テキスト中に記述されている情報を抽出し RDF トリプルにすることで，コンテンツに関する情報を利用できるようにする．コンテンツに関する情報とは，文書中で言及されているエンティティやエンティティに関する言及内容を指す．文書テキストから情報を抽出するには，Entity Linking（EL）や Named Entity Recognition（NER）, Open Information Extraction（OIE）といった既存技術を利用する．EL はテキスト中から外部知識ベースの URI を参照したエンティティとそのクラス情報を抽出する．NER は，固有表現と固有表現から判断されるクラス情報を抽出し，EL で抽出できなかった外部知識ベースにはまだ存在しない新しいエンティティなどの情報を補完する．OIE は，文の構造を解析し，主語・述語・目的語からなる関係をトリプルの形式で抽出する．抽出したエンティティと OIE のトリプルを統合することで，文書中で言及されているエンティティやエンティティに関する言及内容を利用することができる．

統合は，図 3 の破線で囲まれた **2. 文書のコンテンツに関する情報**のように行われる．統合のキーとなる情報はエンティティと OIE のトリプルのオフセット情報である．各オフセット情報が一致する場合，2 つの情報は関係づけられる．具体的には，エンティティのテキスト内での言及位置（オフセット）を URI で表現しリソース（以降，オフセットリソースと呼ぶ）にする．そして，オフセットリソースの目的語としてエンティティの各情報を記述する．オフセット情報が一致する OIE のトリプルは，トリプル全体でエンティティに関する言及内容を表現するため，トリプル全体を 1 つのリソースのように扱う必要がある．そのため，Reification（具体化）[4]と呼ばれる記述方法を OIE のトリプルに適用する．Reification では，トリプル全体を示す rdf:Statement というタイプを付与した空白ノードをつくり，そのプロパティとして rdf:subject，rdf:predicate，rdf:object を与えることで，トリプルの各要素を記述する．そして， OIE トリプル全体を具体化した空白ノードがエンティティのオフセットリソースを言及するようにモデル化することで，エンティティと OIE のトリプルを関係づける．

抽出した情報の記述には，NLP Interchange Format (NIF) [6], Internationalization Tag Set (ITS)[5], Marl Ontology Specification (Marl)[6]のオントロジーの語彙を利用する．これらのオントロジーは自然言語処理や情報システム上での人間による主張をアノテーションするために開発されたものである．

4.1.3　外部の知識ベースの情報

外部の汎用的な知識ベースにあるエンティティの概要情報や分類情報をコンテンツから抽出された情報に付加し，一般的な情報を利用できるようにする．外部の知識ベースの情報は，Entity Linking から得られた外部知識ベースのエンティ

4：https://www.w3.org/TR/rdf-mt/#ReifAndCont
5：https://www.w3.org/TR/its20/
6：http://www.gsi.dit.upm.es:9080/ontologies/marl/

ティ情報を使うことで取得する．取得には，外部知識ベースの SPARQL エンドポイントへのクエリや SPARQL の統合クエリ，知識ベースによって整備されている API などを利用する．本研究では，外部知識ベースの SPARQL エンドポイントに対して，抽出したエンティティのクラス情報やラベルを問合せて取得した．図 3 の **3. 外部の知識ベースの情報**部分のように組み込まれている．

4.2 検索システム

知識ベースに問合せを実行することで，キーワード検索では難しい複雑な検索が可能になる．しかし，知識ベースのデータモデルや語彙を理解した上で複雑なグラフパターン を含む SPARQL クエリを書いて問合せすることは簡単ではない．そこで，図 4 のような検索インターフェースを実装したシステムを提案し，ドメイン知識を持たないカジュアルなユーザでも簡単に検索をできるようにする．検索システムのポイントは次の

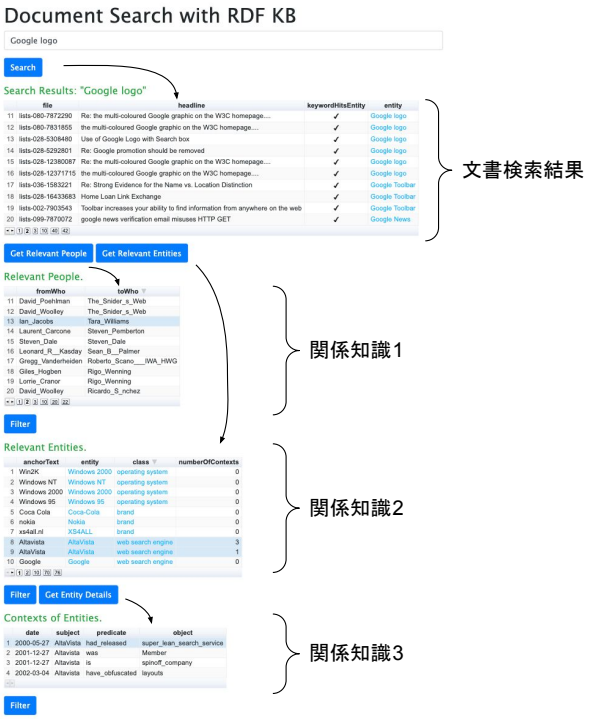

図 4　検索インターフェース

通りである．

（1） 検索キーワードにマッチするエンティティが記載された文書を明示的に提示する（図 4 の文書検索結果）．検索意図に何らかのエンティティが含まれているとき，ユーザはそのエンティティを含む文書を取得したい可能性が高いと考えられる．知識ベースを利用することで，文書一覧にエンティティの有無に関する情報を付加し，ユーザの文書取得を支援する．

（2） 取得した文書一覧の関係知識を表形式で取得する（図 4 の関係知識 1，2）．文書集合に対する知識を持たないユーザは，まずその文書集合に対する理解を深めたいと考えられる．しかし，図 6 の SPARQL クエリ中のグラフパターンが示すように，取得したい情報はマルチホップで関係していることがあり，グラフ形式のまま知識を提示するのはわかりにくい場合がある．そこで，ユーザにとって有用な情報をあらかじめ聴取・定義しておき，必要な情報に絞って表形式で提示する．そして，ユーザは表中の知識を選択することで，その知識と紐づいた実際の文書を取得する．図 4 では，メール文書が誰と誰の間でやりとりされているかや，コンテンツ中で言及されているエンティティの情報を関係知識の例として設定している．

（3） 取得した関係知識に関する知識を取得する（図 4 の関係知識 3）．提示されたエンティティの言及内容を取得することで，そのエンティティが言及された文脈を理解することができる．そして，興味のある文脈で言及されたエンティティを条件にして検索することで，その言及内容が記載された文書を取得することができる．文脈に関するキーワードを文書テキストに適合するように入力することは難しいが，本システムでは検索プロセスの中で，エンティティの文脈を発見することができる．

次に，文書や関係知識を取得するためのモジュールについて説明する．

4.2.1 文書取得モジュール

文書取得モジュールは，キーワードや選択された知識を入力として，文書一覧を出力する．具体的な処理は次の通りである．まず，入力情報から SPARQL クエリを生成する．そして，生成した SPARQL クエリを RDF データベースの SPARQL エンドポイントに要求し，返却された問合せ結果を検索インタフェースに渡す．生成される SPARQL クエリの例が図 5 である．クエリは，ファイル名，文書タイトル，検索キーワードがヒットするエンティティの有無，外部知識ベースのエンティティ情報，を取得する．6, 7 行目の FILTER 句はテキスト検索部分である．9, 10, 11 行目のトリプルパターンは選択された知識を指定している．そして，13-16 行目はエンティティ検索部分である．これらの部分が，入力されたキーワードや選択された知識に応じて生成される．また，複数の知識が選択された場合は，選択された知識ごとに 3 行目から 22 行目を生成し，それらを UNION 句で結合することで全体のクエリを生成する．キーワード検索のみのときは，9, 10, 11 行目のトリプルパターンは生成されない．

4.2.2 関係知識取得モジュール

関係知識取得モジュールは，取得した文書一覧や関係知識を入力として，それらに関係する知識を出力する．ユーザの必要とする知識をあらかじめ定義した上で，開発者は知識取得に必要な SPARQL クエリを設定する．本稿では，取得した文書一覧に関係するエンティティの情報が必要と定義した（図 4 の関係知識 2）．図 6 が対応するクエリである．このクエリは，エンティティのアンカーテキスト，対応する外部知識ベースのエンティティとそのクラス情報，そして，OIE で抽出されたエンティティに関する記述情報の数を取得する．検索インターフェースの “GET Relevant Entities” ボタンが押されると，取得済みの文書一覧がモジュールに入力され，7 行目の FILTER 句が追加される．関係知識も入力された場合は，文書取得モジュールと同様に，選択された知識のトリプルパターンを追加する．生

```
SELECT DISTINCT ?file ?headline (isLiteral(?anchorText) as ?keywordHitsEntity) ?entity ?entityLabel
WHERE{
 ?email schema:alternateName ?file.
 ?email schema:headline ?headline.
 #テキスト検索
 ?email schema:text ?text.
 FILTER(regex(?text,'Google','i')||regex(?text,'logo','i'))
 #関係知識の指定
 ?email schema:mentions ?selectedAnchorText.
 ?selectedAnchorText nif:isString "Altavista"@en.
 ?selectedAnchorText itsrdf:taIdentRef <http://www.wikidata.org/entity/Q433505> .
 #エンティティ検索
 OPTIONAL{
  ?email schema:mentions ?mention.
  ?mention nif:isString ?anchorText.
  FILTER(regex(?anchorText,'Google','i')||regex(?anchorText,'logo','i'))
  #外部知識ベースの情報
  OPTIONAL{
    ?mention itsrdf:taIdentRef ?entity.
    ?entity rdfs:label ?entityLabel.
    }
  }
}ORDER BY DESC (?keywordHitsEntity) DESC (?entityLabel)
```

図 5　キーワード検索 “Google logo” で取得した文書一覧からエンティティ “Altavista” を言及する文書を取得する

成されたクエリは RDF データベースの SPARQL エンドポイントに渡され，返却された問合せ結果を検索インタフェースに渡す．

```
SELECT DISTINCT ?anchorText ?entity ?entityLabel ?class (count(distinct ?triple) as ?numberOfContexts)
WHERE{
 ?email1 schema:mentions ?mention.
 ?mention nif:isString ?anchorText.
 ?email1 schema:alternateName ?file.
 #文書ファイル一覧
 FILTER(?file IN ('lists-080-7872290',...,'lists-002-7747530'))
 #エンティティ情報
 OPTIONAL{
  ?mention itsrdf:taIdentRef ?entity.
  ?entity rdfs:label ?entityLabel.
  ?mention itsrdf:taClassRef ?class.
  }
 #エンティティの言及内容
 OPTIONAL{
  ?triple marl:describesObject ?mention.
  }
 }GROUP BY ?anchorText ?entity ?entityLabel ?class
ORDER BY DESC(?numberOfContexts)
```

図 6　キーワード検索 “Google logo” で取得した文書一覧に含まれるエンティティ情報を取得する

5　評 価 実 験

実験の目的は，特定の検索ニーズにおける知識ベースを利用した高度な文書検索の有用性を評価することである．検索ニーズには，(1) キーワードに関連する文書を調べたい; (2) エンティティx の関連事項に言及する文書を調べたい; (3) エンティティx の○○について言及する文書を調べたい，を設定する．各ニーズに対応する知識ベースを利用した高度な文書検索手法として，(1) 検索キーワードに関係するエンティティの有無を考慮した検索，(2) 外部知識ベースのクラス情報を利用した検索，(3) OIE トリプルの情報を利用した検索，を適用する．以降で，(1), (2), (3) に関する実験結果を報告する．

5.1　データセット

実験で使用するデータセットは，TREC 2005 Enterprise Track[7]のために収集された W3C のメーリングリストのデータセット [8]である．このデータセットは，2005 Email Discussion Search topics において与えられた 60 個のタスクごとに適合文書の評価がされている．本実験においては，そこから 5 つのタスクとそれに対応するデータセットを利用した．実験 1 では与えられている評価を利用し，実験 2，3 では別途設定したタスクに対して人手で文書内容を確認し，正解ラベルを付与した．統計情報を表 1 に示す．

表 1　実験データセットの統計情報

タスク	文書数	実験 1 のタスクの正解文書数の割合	実験 2 のタスクの正解文書数の割合	実験 3 のタスクの正解文書数の割合
3	379	3.4%	3.7%	0.8%
12	434	5.1%	15.0%	2.3%
17	367	2.5%	22.1%	1.9%
21	456	4.4%	37.3%	3.5%
54	372	4.0%	0.5%	0.5%

5.2　提案システム

提案システムでは，文書のコンテンツに関する情報を抽出するために，次のツールを使用した．Entity Linking は TagMe [3]，Named Entity Recognition には TagMe と Spacy[9]，Open Information Extraciton は MinIE [4] である．また，外部の知識ベースは Wikidata [10] を利用している．RDF 化には Python の RDFLib ライブラリ [10]を利用した．RDF ストア（RDF データベース）には，Apache Jena Fuseki 3.12.0 を用いた．検索システムの開発には Node.js を利用した．システムの実行環境は，3.3 GHz dual-core Intel Core i7, macOS Catalina with 16GB RAM である．

5.3　比較システム

比較システムは，全文検索エンジン Apache Lucene 8.7.0 を搭載した Apache Solr 8.7.0 である．Apache Solr 8.7.0 のランキングアルゴリズムは BM25 [8] である．システムの実行環境は，提案システムと同じである．

5.4　実 験 (1)

検索ニーズ 1 に対応するタスクとして，2005 Email Discussion Search topics において与えられたタスクを利用する．具体的なクエリを表 2 に示す．

5.4.1　評 価 内 容

比較システムの Apache Solr において，与えられたクエリで取得した文書一覧をベースライン（Solr）として，クエリのキーワードに部分一致するエンティティを含む文書のランク値を 1.2 倍した結果（Solr+entity）を比較した．なお，ストップワード

7 : https://trec.nist.gov/data/t14_enterprise.html
8 : https://tides.umiacs.umd.edu/webtrec/trecent/parsed_w3c_corpus.html
9 : https://spacy.io/
10 : https://rdflib.readthedocs.io/en/stable/

表 2　実験 1 のクエリ

タスク	クエリ
3	Google logo
12	Middleware compared to CGI and Perl combinations
17	Application layer API to IPsec
21	Signature portability
54	Webmail vulnerabilities

（to, and）はクエリから除去した．評価指標は， Precision@R (R=10, 20, 30) である．

5.4.2　結　　果

結果は表 3 の通りである．全体の傾向として，検索キーワードに部分一致するエンティティの有無を考慮することで，ランキングの精度が向上することを確認できた．21 は，ベースラインの適合率も低いことに加えて，他のクエリに比べて抽象的なキーワードから構成されるクエリであるため，具体的なエンティティの有無の考慮が検索精度の向上に寄与しなかったと考えられる．

表 3　実験 1 の結果

タスク	手法	Precision@10	Precision@20	Precision@30
3	Solr	0.00	0.25	0.23
3	Solr+entity	**0.50**	**0.35**	0.23
12	Solr	0.40	0.30	0.37
12	Solr+entity	**0.60**	**0.55**	**0.40**
17	Solr	0.00	0.20	0.17
17	Solr+entity	**0.40**	**0.25**	**0.20**
21	Solr	0.00	**0.10**	**0.07**
21	Solr+entity	0.00	0.00	0.00
54	Solr	0.00	0.00	0.07
54	Solr+entity	**0.50**	**0.25**	**0.17**

5.5　実　験　(2)

検索ニーズ 2 に対応するタスクとして，実験 1 の取得結果を基に追加で検索を行うタスクを設定した．詳細を表 4 に示す．

5.5.1　評 価 内 容

比較システムではクエリ拡張を利用した検索を行い，提案システムではクラス情報を利用した検索を行なった．比較システムでは，実験 1 のクエリのキーワードと表 5 で示すクエリ拡張キーワードの AND 検索を行なった．クエリ拡張キーワードは，表 5 のクラス情報に属すインスタンスやサブクラスのラベルに展開されて検索が実行される．提案システムの検索は，システム上で確認できた表 5 のクラス情報をもつエンティティを選択することで生成される SPARQL クエリによって実行された．検索結果は Precision と Recall で評価した．

表 4　実験 2 のタスク

タスク	タスク内容
3	他の検索エンジンについて言及している文書を調べる
12	他のプログラミング言語について言及している文書を調べる
17	暗号プロトコルの種類（サブクラス）について言及している文書を調べる
21	デジタル署名の種類（サブクラス）について言及している文書を調べる
54	ウェブメールクライアントについて言及している文書を調べる

表 5　実験 2 のクエリ

タスク	クエリ拡張（拡張キーワード数）	クラス情報
3	search engine (75)	web search engine (Q4182287)
12	programming language (1421)	programming language (Q9143)
17	cryptographic protocol (16)	cryptographic protocol (Q1254335)
21	digital signature (9)	digital signature (Q220849)
54	web mail client (12)	webmail (Q327618)

5.5.2　結　　果

結果は表 6 の通りである．全体的な傾向として，提案手法は Recall よりも Precision が高くなる結果を示した．提案手法ではエンティティリンキングによって，文字列の一致だけでなく文脈を考慮した意味の一致を検証している．したがって，抽出されたエンティティに必要とするクラスが付与されている場合，ほぼ正確に適合文書を検出できるが，適切なエンティティが抽出されていない場合や，異なるクラスが付与されている場合は検出ができず，Recall が低くなった．一方，比較手法の結果を分析すると，不適合文書の多くは，単語の羅列からなるスパムメールや別の意味で使われている同形異義語（homograph）を含むメールであった．クエリ拡張では意味的な検証はなく，文字列の一致のみを検証していることが原因であると考えられる．また，同じ意味を示すキーワードであっても，クエリ拡張のキーワード文字列が文書中の文字列に一致しない場合は，適合文書を検出することができないため，必ずしも Recall が高くならなかった．

表 6　実験 2 の結果

タスク	手法	Precision	Recall	取得文書数
3	比較手法	0.44	**0.85**	25
3	提案手法	**0.73**	0.79	15
12	比較手法	0.18	0.26	95
12	提案手法	**0.93**	**0.58**	41
17	比較手法	**1.00**	0.25	20
17	提案手法	0.94	**0.42**	36
21	比較手法	0.87	**0.72**	140
21	提案手法	**0.99**	0.39	67
54	比較手法	**0.67**	**1.00**	3
54	提案手法	0.50	0.50	2

5.6　実　験　(3)

検索ニーズ 3 に対応するタスクとして，表 7 で示すタスクを設定した．

表 7　実験 3 のタスク

タスク	タスク内容
3	Google の Web サイト上のリンクに対する考え方に言及した文書を調べる
12	CGI のインターフェースについて言及した文書を調べる
17	IPsec のハッシュ関数について言及した文書を調べる
21	デジタル署名による XML 検証について言及した文書を調べる
54	プロキシのキャッシュの脆弱性について言及した文書を調べる

5.6.1　評 価 内 容

比較システムではタスクから想起されるキーワードをクエリとして検索を行なった．提案システムでは，タスクに関係するエンティティの記述（OIE のトリプル）を選択して生成される SPARQL クエリによって検索を行なった．具体的な内容は表 8 の通りである．検索結果は Precision，Recall で評価した．

表 8　実験 3 のクエリ

タスク	比較手法のクエリ	提案手法のクエリ (OIE トリプル)
3	Google AND (think OR consider) AND link	(Google, consider,“link text”)
12	CGI AND interface	(CGI, has, “form interface”)
17	IPsec AND hash function	(IPSEC, is_specifing_hash_function_for, “authentication purposes”)
21	XML validation AND digital signature	(XML_validation, works_with, “XML Digital Signatures”)
54	Caching proxies AND vulnerabilities	(Caching_proxies, provide, “additional potential vulnerabilities”)

5.6.2 結　　果

結果は表 9 の通りである．提案手法では，Precision が 1 で，OIE トリプルで示されたエンティティの言及内容を利用することで，正確に適合文書を取得することができた．一方で，複数文に渡って言及しているような他の適合文書は取得することができなかったため，Recall が低くなった．比較手法では，Recall が高く，Precision が低くなった．検索キーワードにはマッチするが，タスクを満たさない不適合文書の取得が多くなったことで，Precision の低下に至ったと考えられる．以上より，提案手法は確実に適合文書を取得するのに有効だといえる．

表 9　実験 3 の結果

タスク	手法	Precision	Recall	取得文書数
3	比較手法	0.14	**1.00**	21
3	提案手法	**1.00**	0.33	1
12	比較手法	0.17	**0.60**	36
12	提案手法	**1.00**	0.10	1
17	比較手法	0.43	**0.86**	14
17	提案手法	**1.00**	0.14	1
21	比較手法	0.11	**0.56**	81
21	提案手法	**1.00**	0.13	2
54	比較手法	**1.00**	**1.00**	2
54	提案手法	**1.00**	0.50	1

6　まとめと今後の課題

本稿では，ある文書集合に詳しくないユーザでも，効率的に興味のある文書を調べることができるようにすることを目的として，(1) 文書集合に関する知識を構造化した RDF 知識ベースの構築と，(2) 知識ベースに対する高度な検索を簡単にするための探索的な検索システム，を提案した．提案する知識ベースのモデルは文書の内容に関するオントロジーを必要としないため，構築にかかる負担が少なく，さまざまな文書への応用が期待できる．評価実験は，現実世界のデータセットを用いて，一般的に広く使われている全文検索システムとの Precision と Recall に関する比較を行なった．結果は，検索キーワードに関係する文書中のエンティティ有無を考慮することで，文書検索における Precision の精度を良化させることを示した．また，知識ベース中のクラス情報やエンティティに関する言及内容を利用した検索は，Recall よりも Precision において優れた精度を示した．

今後の課題として，被験者実験による検索システムの評価が挙げられる．また，エンティティの有無を考慮したより洗練されたランキングアルゴリズムの検討や，より大規模なデータセットへの適用が考えられる．

7　謝　　辞

この成果は，国立研究開発法人新エネルギー・産業技術総合開発機構（NEDO）の委託業務（JPNP20006）の結果得られたものです．

文　　献

[1] Ravish Bhagdev, Sam Chapman, Fabio Ciravegna, Vitaveska Lanfranchi, and Daniela Petrelli. Hybrid search: Effectively combining keywords and semantic searches. In *European semantic web conference*, pp. 554–568. Springer, 2008.

[2] Paul Alexandru Chirita, Rita Gavriloaie, Stefania Ghita, Wolfgang Nejdl, and Raluca Paiu. Activity based metadata for semantic desktop search. In *European Semantic Web Conference*, pp. 439–454. Springer, 2005.

[3] Paolo Ferragina and Ugo Scaiella. Tagme: on-the-fly annotation of short text fragments (by wikipedia entities). In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pp. 1625–1628, 2010.

[4] Kiril Gashteovski, Rainer Gemulla, and Luciano del Corro. MinIE: Minimizing facts in open information extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 2630–2640, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[5] Rasmus Hahn, Christian Bizer, Christopher Sahnwaldt, Christian Herta, Scott Robinson, Michaela Bürgle, Holger Düwiger, and Ulrich Scheel. Faceted wikipedia search. In *International Conference on Business Information Systems*, pp. 1–11. Springer, 2010.

[6] Sebastian Hellmann, Jens Lehmann, Sören Auer, and Martin Brümmer. Integrating nlp using linked data. In *International semantic web conference*, pp. 98–113. Springer, 2013.

[7] S. Oramas, Luis Espinosa Anke, M. Sordo, Horacio Saggion, and X. Serra. Information extraction for knowledge base construction in the music domain. *Data Knowl. Eng.*, Vol. 106, pp. 70–83, 2016.

[8] Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. Okapi at trec-3. *Nist Special Publication Sp*, Vol. 109, p. 109, 1995.

[9] Andon Tchechmedjiev, Pavlos Fafalios, Katarina Boland, Malo Gasquet, Matthäus Zloch, Benjamin Zapilko, Stefan Dietze, and Konstantin Todorov. Claimskg: a knowledge graph of fact-checked claims. In *International Semantic Web Conference*, pp. 309–324. Springer, 2019.

[10] Denny Vrandečić and Markus Krötzsch. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, Vol. 57, No. 10, pp. 78–85, 2014.

[11] Jörg Waitelonis and Harald Sack. Towards exploratory video search using linked data. *Multimedia Tools and Applications*, Vol. 59, No. 2, pp. 645–672, 2012.

[12] An-Zi Yen, Hen-Hsen Huang, and Hsin-Hsi Chen. Personal knowledge base construction from text-based lifelogs. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 185–194, 2019.

[13] Sivan Yogev, Haggai Roitman, David Carmel, and Naama Zwerdling. Towards expressive exploratory search over entity-relationship data. In *Proceedings of the 21st International Conference on World Wide Web*, pp. 83–92, 2012.

[14] Moshé M Zloof. Query by example. In *Proceedings of the May 19-22, 1975, national computer conference and exposition*, pp. 431–438, 1975.