原著論文 Original Paper

Linked Data統合に向けたLiteral値マッチング手法の提案

Proposal of Literal Matching Method toward Linked Data Integration

川村 隆浩 (株) 東芝 研究開発センター
Takahiro Kawamura Corporate Research & Development Center, Toshiba Corp.

長野 伸一 （同 上）
Shinichi Nagano

大須賀 昭彦 電気通信大学 大学院情報システム学研究科
Akihiko Ohsuga Graduate School of Information Systems, University of Electro-Communications, Japan

keywords: linked open data, literal matching, instance matching

Summary

Linked Open Data (LOD) has a graph structure in which nodes are represented by URIs, and thus LOD sets are connected and searched through different domains. In fact, however, 5% of the values are literal (string without URI) even in DBpedia, which is a *de facto* hub of LOD. Therefore, this paper proposes a method of identifying and aggregating literal nodes in order to give a URI to literals that have the same meaning and to promote data linkage. Our method regards part of the LOD graph structure as a block image, and then extracts image features based on Scale-Invariant Feature Transform (SIFT), and performs ensemble learning, which is well known in the field of computer vision. In an experiment, we created about 30,000 literal pairs from a Japanese music category of DBpedia Japanese and Freebase, and confirmed that the proposed method correctly determines literal identity with F-measure of 76–85%.

1. はじめに

本研究の目的は，$< Resource, Property, Value >$の3項からなるLinked Dataにおいて，Literal（IRIなしの文字列）で表された値を可能な限り結合（リソース化）し，本来のリンクされたデータの形とすることである*1．現在，事実上，全世界のLODのハブとなっているDBpedia（Wikipediaの一部をLOD化したもの）においても，我々の調査によればValueの約5%はLiteral値となっており，検索する際にはリンクを辿ることができず，正規表現マッチなどの手法に頼る必要がある．また，我々が現在進めているWeb情報やソーシャルデータからのLOD生成では，元データが文章であるため，少なくとも初期段階では数多くのLiteral値が生成される．今後，世界規模で異種LODを結合してLODクラウドを形成していくにあたってはより大きな問題となるだろう [Bizer 09]．そこで，同一の内容を表すと思われるLiteral値を判別し，リソース化を助けるために，LODのグラフ構造を利用したLiteral値のマッチング手法を提案する．これは，オントロジーアライメント分野におけるインスタンスマッチングに近い問題であるが，インスタンス（LODではリソース）ではなく末端ノードにあたるLiteral値のマッチングにフォーカスしている点が異なっている．また，システム・インテグレーション（SI）案件において従来からある名寄せ問題と類似している．尚，本論文で提案する手法は，LOD上の対象Literal値周辺の構造を画像として捉え，対象Literal値を特徴点とした画像特徴量を抽出し，類似画像の判別という考え方でLiteral値の同一性を判定しているところに特徴があり，本論の主な貢献は本問題に向けた新たな特徴量を提案している点にある．

以下，2章で関連研究を示し，3章でLODからの画像特徴量の抽出について提案する．そして，4章でアンサンブル学習を用いてLiteral値の同一性を判定した結果を，単純な文字列マッチングの場合として比較して評価する．最後に，5章でまとめと今後の進め方について述べる．

*1 W3Cの定義によれば，リソースにはIRIとLiteralの両方が含まれるが，本論では区別するためIRIのみを指してリソースと呼称する．

2. 関 連 研 究

本論で取り上げる LOD における Literal 値のマッチングに関しては，残念ながらこの問題を中心トピックとして取り組んだ研究は見つかっていない．[Raimond 08, Hogan 06, Wielemaker 08] などが，インスタンスマッチングの過程において Literal 値のマッチングに触れているが，いずれも Literal 値をキーワードや正規表現で検索した後に，編集距離や頻度でソートする手法が用いられている．そこで，本章ではインスタンスマッチングに関する研究を一部紹介し，最後に Literal マッチングとの違いについて述べる．

インスタンスマッチングにおいて，よく知られたツールとして SILK[Volz 09a, Volz 09b] が挙げられる．SILK では，ユーザが類似性を判定する特徴量を選択したり，データセット毎にリンクの仕様を定義することができる．更に，Maali らは Google が提供するデータ整形ツール Google Refine[Google 14] 上に，

(1) SPARQL 検索
(2) キーワード検索付き SPARQL 検索
(3) SILK サーバー連携
(4) キーワードによる Sindice[Sindice 14] 検索と (2) の組み合わせ（Sindice はキーワードがマッチする RDF ドキュメントの URI を返すサービスであるため）

の 4 つの Extension を実装し，それぞれの精度とパフォーマンスを計測している [Maali 11]．Literal マッチングに共通する多くの知見が含まれているが，いずれも比較対象となる 2 つのデータセットのデータ構造を予め知っていないと検索式や変換ルールを書けないという問題が存在する．そこで，データセットに対する事前知識を不要とするため，Genetic Programming を用いて自動的にルールを生成する研究も行われている [Isele 11]．同様のアプローチは，[Niu 12] にも見られる．ここでは，半教師あり学習を用いてデータセット毎のマッチングルールを獲得，繰り返しリファインしていくアプローチが取られている．その他の研究も大きくはドメイン（データセット）毎の知識に依存して精度を上げるか，ドメイン毎の教師データ作りなどを回避するためにドメイン非依存を目指すかに分けることができるだろう．

前者の 1 つに Rong らの研究 [Rong 12] が挙げられる．ここではインスタンスマッチングを 2 値分類問題として定義し，学習によって判別している．但し，ドメインに依存した教師データを少なくするために転移学習 TrAdaBoost を用いている点に特徴がある． 他にも，ObjectCoref[Hu 10] では，教師データを少なくするために特徴的なプロパティ，値のペアを繰り返し見つける自己学習フレームワークを導入している．いずれも本稿で提案する手法の発展として参考にしたい．

一方で，後者の 1 つに RiMom[Li 09] が挙げられる．ドメイン非依存なアプローチは学習を行わず，String Similarity（Jaccard 係数，cosine，TF/IDF，編集距離など），Semantic Similarity（WordNet や Inverse Functional Property など），Structural Similarity など複数の特徴量を組み合わせるものが多い．RiMom では，マッチング対象データセットに合わせて自動的に特徴量の組み合わせを提案する．

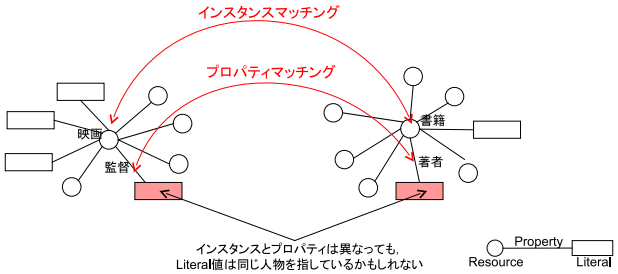

図 1 関連研究との関係

他にも，インスタンスマッチングに関しては [Castano 11] のサーベイが参考になるだろう．これらインスタンスマッチングと Literal 値のマッチングは類似した問題と見なせるが，Literal 値のマッチングは，インスタンス（リソース）化されていない複数のノードが同じ実体を指していれば，それらをマッチングさせるものであり，インスタンスマッチングが行われ，プロパティのマッチング [Gunaratna 13] が行われた後にも残る問題である．例えば，映画リソースと書籍リソースの “監督” プロパティと “著者” プロパティの指す人物が同じ，という場合などである．その場合，インスタンスマッチングをしても Literal 値の同一性は発見され得ない．インスタンスマッチング，プロパティマッチング，Literal マッチング三者の関係を図 1 に示す．そこで，我々はインスタンスマッチングにも使われる要素技術を参考にしながら，独自の Literal マッチング手法を考案した．

3. Literal マッチングの問題定義と特徴量の抽出

本章では，まず Literal マッチングを 2 値分類問題として定義した上で，分類器にかけるための特徴量を抽出する手法を提案する．

3·1 2 値分類問題の定義

本論では，先行研究 [Rong 12] をベースに，Literal マッチング問題を以下の 2 値分類問題として定義する．

定義. 異なる *Literal* 値 a, b が，実世界において同じオブジェクトを指している場合に $(a, b) \in \mathcal{R}$ と表す．*Literal* 値のマッチング問題とは，*2* つの *LOD* グラフ *A, B* が与

えられた際に以下の集合 $\mathcal{M}$ を得ることである．

$$\mathcal{M} = \{(a,b)|(a,b) \in A \times B, (a,b) \in \mathcal{R}\}$$

また，同一の *Literal* 値のペアを見つける問題は，以下のような *2* 値分類問題として表される．

$$\mathcal{C} : (a,b) \rightarrow \{0,1\}\ for\ (a,b) \in A \times B$$

ここで，$\mathcal{C}$ はマッチしないペアに *0* を，マッチするペアに *1* を割り当てる分類器である．

3·2 グラフ構造からの画像特徴量の抽出

次に，分類器に入力する特徴量を生成するため，画像分類の研究においてよく用いられる SIFT（Scale-Invariant Feature Transform）[Lowe 99] をベースとした特徴量の生成手法を提案する．SIFT とは，特徴点毎に算出される局所特徴量であり，特徴点の周辺領域を一辺 4 マスの計 16 マスに分割し，それぞれ 8 方向 (45 度ずつ) の輝度勾配を 128 次元のベクトルとする手法である．

図 2 に，2 つ LOD グラフからブロック画像を生成する手法の概要を示す．ここでは，2 つの比較対象 Literal 値に繋がるプロパティ, リソース，および同リソースに繋がる他 2 つのプロパティとその値を選択し，それぞれの類似度 Sim_l, Sim_p, Sim_r を計算することで 3×3 マスからなるグレースケール画像を生成する．各マスは $[0,1]$ の値を持ち，1 に近いほど黒く表現されている．尚，他 2 つのプロパティと値に関しては，対象リソース間で共通プロパティが存在する場合は，それらをアルファベット順に選択し，リソース，プロパティ，値をそれぞれ第 2 行の第 1 列，第 2 列，第 3 列に，および第 3 行に同様に割り当てる．共通プロパティが存在しなかった場合は，以下の計算式で最も類似度の高かったプロパティを 2 つ選択して同様に割り当てることとし，一定の規則で画像が作られるようにする．類似度 Sim_l, Sim_p, Sim_r は，String Similarity と Semantic Similarity の組み合わせとして以下のように定義する．

$$\begin{aligned} Sim_l(l_{1i}, l_{2j}) =& \alpha\, StringSim(l_{1i}, l_{2j}) \\ &+ \beta\, SemanticSim(l_{1i}, l_{2j}) \end{aligned} \tag{1}$$

$$\begin{aligned} Sim_p(p_{1i}, p_{2j}) =& \gamma\, StringSim(p_{1i}, p_{2j}) \\ &+ \delta\, SemanticSim(p_{1i}, p_{2j}) \end{aligned} \tag{2}$$

$$\begin{aligned} Sim_r(r_1, r_2) =& \epsilon\, StringSim(r_1, r_2) \\ &+ \zeta\, SemanticSim(r_1, r_2) \end{aligned} \tag{3}$$

$$StringSim(S1, S2) = \frac{|S1 \cap S2|}{|S1 \cup S2|} \tag{4}$$

$SemanticSim(S1, S2) =$

$$\begin{cases} 1.0 & \text{if} & \text{S1 = S2} \\ 0.75 & \text{else if} & \text{S1 is } \textit{Synonym} \text{ of S2} \\ 0.5 & \text{else if} & \text{S1 is } \textit{Hypernym} \text{ or } \textit{Hyponym} \text{ of S2} \\ 0.25 & \text{else if} & \text{S1 has same } \texttt{namespace} \text{ as S2} \\ 0.0 & \text{else} & \end{cases} \tag{5}$$

l_{ij}, p_{ij}, r_{ij} は，それぞれ Literal ノード，プロパティ，リソースの ID である．また，$\alpha, \beta, \gamma, \delta, \epsilon, \zeta$ は係数であり，$0 \leqq \alpha, \beta, \gamma, \delta, \epsilon, \zeta, \alpha+\beta, \gamma+\delta, \epsilon+\zeta \leqq 1$ を満たすものとする．次章では，対象ドメインに依存しない中立的な設定として，重みがいずれかに偏らないようにいずれも 0.5 に設定している．$StringSim$ は Jaccard 係数，$SemanticSim$ はオントロジーマッチング研究におけるクラス間のパス長を利用した類似度の指標を簡略化したものに相当する．尚，*Synonym*，*Hypernym*，*Hyponym* の判定にはオントロジーを参照しており，次章では WordNet[*2]，および日本語 WordNet[*3]を用いている．

本画像の直感的な意味合いは，文脈に類似したグラフ上の脈を表している点にある．つまり，グラフ上における意味内容の繋がり具合いを画像として表している．例えば，異表記の同義語であれば図 3(a) のように基準点の絶対値を除いて同じような類似度を持つ画像となる．一方，同表記の異義語の場合，図 3(b) のように基準点は同じでも周辺の類似度が異なる画像として表されることが期待される．このように特徴点（対象 Literal 値）の周辺リンク，ノードの類似性を画像として表現することで，グラフ内での意味の繋がりを特徴付けている．尚，本手法はインスタンスやオントロジーマッチングにおいて木構造を比較する手法の変形であるとも言える．但し，Literal ノードは末端であるため，グラフ上，木としては捉えられない．そこで，共通プロパティ，類似プロパティに繋がる周辺のノードを含めてブロック画像を構築している．

最後に，作成したブロック画像において対象 Literal 値を特徴点とし，隣接する 9 マスの画像の類似度の変化を特徴量とする．但し，SIFT と異なり方向性は設定しない．また，特徴点そのものは類似度の絶対値を入れ，基準点としている．特徴量 $\boldsymbol{v}$ は以下の式によって表される．尚，η, θ, λ は係数であり，次章では対象ドメインに依存しない中立的な設定として，いずれも 1.0 に設定している．

∗2 wordnetweb.princeton.edu/perl/webwn
∗3 nlpwww.nict.go.jp/wn-ja

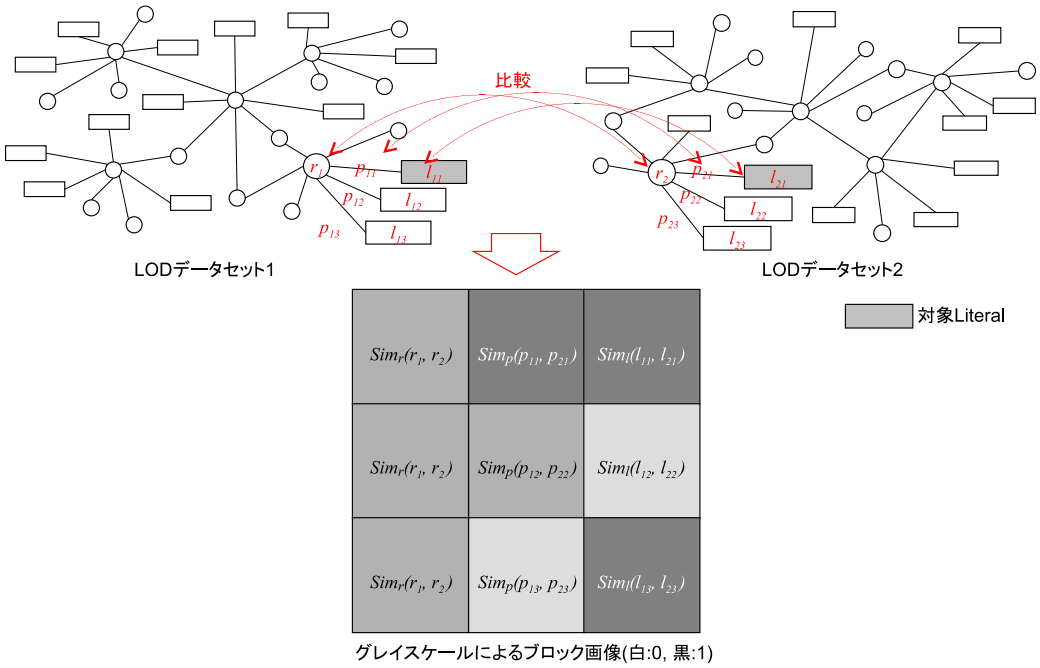

図 2 LOD からの画像生成

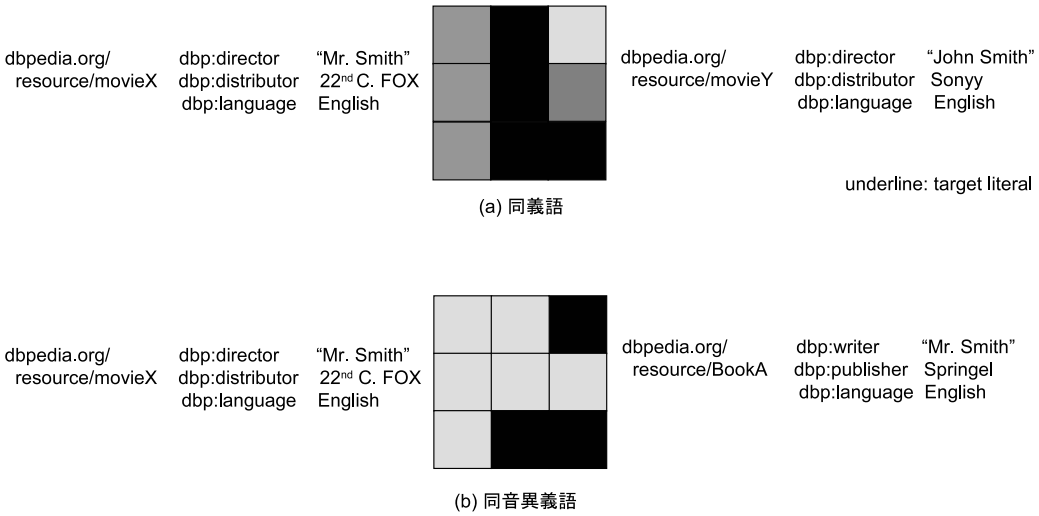

図 3 同義語，同音異義語の特徴量の例

$$
\begin{aligned}
\boldsymbol{v} =& Sim_l(l_{11}, l_{21}) + \Delta_{p1} + \Delta_{r1} \\
& \Delta_{l2} + \Delta_{p2} + \Delta_{r2} \\
& \Delta_{l3} + \Delta_{p3} + \Delta_{r3} \\
\Delta_{li} =& \eta \left(Sim_l(l_{1i}, l_{2i}) - Sim_l(l_{1(i-1)}, l_{2(i-1)}) \right) \\
\Delta_{pi} =& \theta \left(Sim_p(p_{1i}, p_{2i}) - Sim_l(l_{1i}, l_{2i}) \right) \\
\Delta_{ri} =& \lambda \left(Sim_r(r_1, r_2) - Sim_p(p_{1i}, p_{2i}) \right) \quad (i = 1, 2, 3)
\end{aligned}
\tag{6}
$$

SIFT において勾配（輝度変化）を取るのは，照明の違いなどによる絶対値の違いを吸収するためであるが，LOD セットの比較においても，同じスキーマ（プロパティ定義）を用いたデータセット同士であれば，同プロパティは完全一致する上，一定の規則に基づいて設計されたプロパティ名称であれば類似プロパティ同士についても正しく類似性が判定されることで，マッチするペア周辺のリンク・ノードの類似度は全体的に高い値になると予想される．それに対して，異なるスキーマを用いたデータセット同士の比較では，全体的にそれより低い値になると予想した（図 4 参照）．実際に次章で示す実験データの例では，同じスキーマを用いたデータセット (DBpedia) 同士の比較における類似度の絶対値の平均は 0.40，異なるスキーマを用いたデータセット (DBpedia と Freebase) の比較における類似度の絶対値の平均は 0.27 という結果であった．それぞれの標準偏差は 0.39，0.31 となっており，単純な比較は難しいが分布に大きな差がないことなどから，類似度のレベルが相対的に異なっていると判断した（但し，これはブロック画像の類似度の変化がそのままシフトしているという意味ではない）．そこで，提

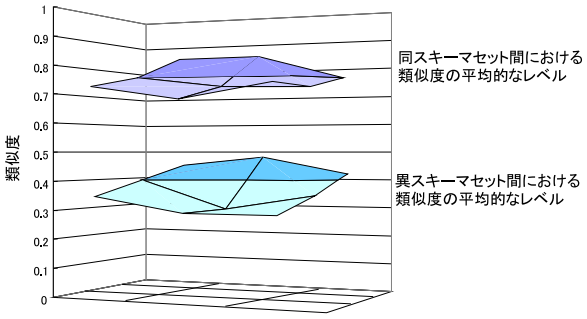

図 4 セット間の類似度の相対的なレベルの差（予想）

案手法においては類似度の絶対値の差を吸収し，より汎用的な特徴量とするために，Sim の絶対値ではなく，隣接する点からの差分（類似度の勾配）を学習対象とした．次章 4·4 節にて，この効果についても確かめる．

4. Literal ペアの分類実験

本章では，提案手法の有効性を検証するために行った実験結果を示す．既存手法と比較するにあたり，LOD の Literal マッチングに関しては筆者らの調べた限り，関連研究冒頭で述べた単純な手法を除いて比較対象となる手法またはコードを見つけることができなかった．しかし，既存のインスタンスマッチング手法を変形して Literal マッチングに適用した場合，それ自体，また別のオリジナルな手法となってしまうだろう．そこで，企業の SI 業務において名寄せ等のデータクレンジングによく用いられる Apache Solr[Apache 14] をベースライン手法として比較を試みた．

4·1 実 験 設 定

まず，同スキーマのデータセット間の Literal マッチングを行うため，DBpedia Japanese の “JPOP” カテゴリ以下のリソース群から Literal 値を持つ計 $N = 8{,}504$ トリプルを抽出し，$(N^2 - N)/2 = 36{,}154{,}756$ ペアを生成した．そして，その中から明らかにマッチングさせる必要のないペア（時間と場所など）を削除して 17,391 ペアを選択した上で，マッチングの有無について人手で正解値を設定した．尚，ペアの削除と正解付けは，研究員 2 名が全ペアを目視で確認し，意見が分かれた場合は協議により決定した．但し，判断基準が簡単（同じ内容を指す文字列かどうか）であったため，協議に至るケースはほとんどなく，その内の多くは一方が対象文字列に関する知識を持っていない場合であったため，協議により問題なく解決した．同一と見なされた Literal 値のペアの例としては，リソースの `label` や `alias` の表記ゆれ，リソース化されていないアルバム名やメンバー名の表記ゆれや異表記，記録情報（「オリコン 1 位」など），住所，時間表記などもリソース化されていないものが見られた．尚，同一ペアは全体の 2.78%であった．この内，2 ペア (0.01%) を除く全てが 2 章で問題としたインスタンスまたはプロパティが異なり Literal 値が同じとなるペアであった．

また，異なるスキーマのデータセット間のマッチングを行うため，LOD 版 Freebase の “J-POP” カテゴリ以下のリソース群から Literal 値を持つ計 $M = 2{,}879$ トリプルを抽出し，上記の DBpedia から抽出したトリプル $N = 8{,}504$ と組み合わせて，$N * M = 24{,}483{,}016$ ペアを生成した．そして，上と同様に明らかにマッチングさせる必要のないペアを削除して 13,702 ペアを選択し，マッチングの有無について人手で正解値を設定した．尚，ペアの削除と正解付けも同様に，研究員 2 名による全ペアの目視と協議により決定した．しかし，より大量の問題への正解付け，またはより複雑な関係を判断する場合は，研究員だけで処理することは難しく，また協議によっても意見が分かれるケースが現れることが予想される．こうした正解付けの問題は機械学習の分野で以前より指摘されており，昨今ではヒューマンコンピュテーション（人間を計算資源として取り込むアプローチ）や，クラウドソーシング（不特定多数への業務委託）を用いた手法が広く検討されている．こうした手法においては，大量問題の処理だけでなく，ユーザへのタスク（聞き方）を如何に設定するかで意見が分かれやすい問題をどう処理するかなども研究されている．著者らも独自のクラウドソーシングプラットフォーム [芦川 14] を運用してきており，今後，本問題への適用を検討していきたい．尚，同一ペアは全体の 1.55%であった．これら全てが 2 章で問題としたインスタンスまたはプロパティが異なり Literal 値が同じとなるペアであった．

次に，上記の各ペアに関して前章で示した画像特徴量を算出し，分類器としてアンサンブル学習の 1 つ，Random Forest[Breiman 01] を用いて学習させ，10 交差検定法によって精度評価を行った．アンサンブル学習とは，弱学習器を組み合わせて高精度の学習器を構成する手法であり，バギングやブースティング，Random Forest といったアルゴリズムが存在する．特に，Random Forest は決定木ベースの集団学習アルゴリズムであり，説明変数への依存が少ないことや学習が高速であることが特徴として挙げられている．インスタンスマッチングを扱った他の研究 [Rong 12] でも，問題の性質から Random Forest の性能がよいことが示されている．実際，以下の実験においても SVM(Support Vector Machine) を用いた場合，学習データへの依存が強く，適合率は上がるものの，再現率が下がる傾向が確かめられている．尚，決定木の数は 10，それぞれランダムに 4 つの特徴量を用いており，分岐の深さに制限は設けていない．また，アルゴリズムは C4.5 をベースとした実装（modified REPTree）である．処理の流れを図 5 に示す．

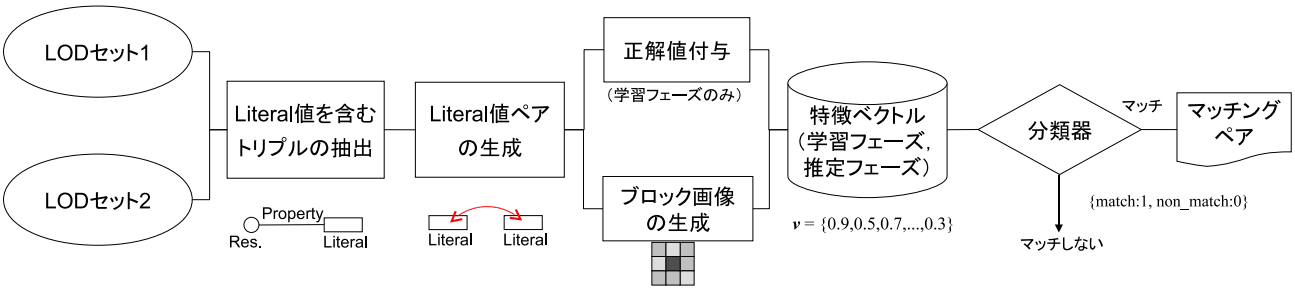

図 5 Literal マッチングの処理の流れ

4·2 ベースライン手法と提案手法の比較

まず，同スキーマのデータセットに対して，ベースライン手法と提案手法を適用した結果を示す．ベースライン手法としては，Apache Solr を用いた検索結果を用いた．

Apache Solr とは，オープンソースの全文検索エンジンである．2014 年現在，コミュニティの活発さ，更新頻度，ダウンロード数，Twitter 社等 Web 系企業への導入実績から，世界で最も利用されているオープンソース検索エンジンと言えるだろう．また，Solr におけるトークナイザ（N-gram, 日本語形態素解析）と各種フィルタは，検索語の表記ゆれ対策や名寄せ等を行うデータクレンジング機能としてしばしば用いられている [石井 12, 酒井 13]．トークナイズに用いられる形態素解析エンジンは Kuromoji[Kuromoji 14]，標準辞書は IPA-dic である．フィルタは，トークナイザによって分割された文字列に対して，動詞や形容詞等の辞書形への変換，特定品詞の除外（助詞など），半角全角変換，大文字小文字変換，ストップワードや長音記号の除去などを行う（各フィルタのオンオフは設定ファイルによって変更できるが，以下の実験ではデフォルトの設定，全てオンを用いている）．登録文書は，上記のトークナイズ，フィルタ処理が行われた後，転置インデックスが作成され，格納される．検索時には，検索語も同様にトークナイズ，フィルター処理が行われた後，TFIDF のベクトルモデルに基いて登録文書と比較，スコアリングされて，最もマッチすると思われる順に登録文書が返される．ベースライン手法では，各 Literal 値を登録文書と見なして Solr に事前に登録，インデックス化しておき，一方の Literal 値によって，対となるもう一方の Literal 値が検索されるかどうかで，マッチするかどうかを判断した．尚，Solr には類義語辞書の検索機能も付いているが，類義語は手動で登録していく必要があり（デフォルトでは空である），比較条件の統一の意味でも今回は用いていない．マッチング判定結果を表 1 に示す．表中，TRUE はマッチするペアを，FALSE はマッチしないペアを表す．

結果として，FALSE に関しては適合率，再現率ともに高い値を示した．全ペア中，Literal 値が同一のペアは 3%未満であることから，マッチングを厳しく判定し，圧倒的多数の FALSE の精度を上げれば，トータルでの精度を上げるのは容易な問題であることが読み取れる．また，

表 1 ベースライン手法による判定結果 (%)

	$3-best$		$1-best$	
	Precision	Recall	Precision	Recall
TRUE	64.7	57.8	79.4	49.5
FALSE	98.8	99.1	98.6	99.6

TRUE に関しては 1-best と 3-best（上位 3 つ以内に該当する語が含まれていれば正解とする）を比較すると分かるように，マッチングに厳密性を求めれば TRUE の適合率を上げるのは容易である．しかし，結果として再現率が低下し，取りこぼしが多くなっている．つまり，Literal 値のマッチングにおいては，TRUE と FALSE の結果を分けて評価し，TRUE の再現率を下げずに適合率を上げることが課題であることが分かる．

次に，本提案手法による判定結果を表 2 に示す．

表 2 画像特徴量による判定結果（同スキーマの場合）(%)

	Precision	Recall	F-Measure
TRUE	87.5	82.6	85.0
FALSE	99.5	99.7	99.6
Weighted Avg.	99.2	99.2	99.2

結果として，提案手法では TRUE の適合率が 1-best の場合に比べ 8 ポイント程度向上させながら，再現率は 3-best の場合よりも 25 ポイント程度向上させていることが確認できた．ベースライン手法から改善のあったの例としては，愛称問題など（“スターダスト・レビュー” と “スタ★レビ” 等）が挙げられる．これは図 3(a) で示したように，基準点の類似度は低いが，周辺リンク・ノードの高い類似度が反映されたためと思われる．しかし，同じ愛称問題でも “寺田光男” と “つんく♂” のように基準点の類似度が 0 となってしまう場合は，TRUE の確率が 0.23 と算出され，FALSE と判定されている（失敗している）．また，FALSE を含めたペア数に応じた加重平均では，適合率，再現率，F 値ともに 99%超という結果を得ることができた．尚，OOB (out-of-bag) エラーは約 1%であった．この結果から，本手法の有効性を確かめることができた．

4·3 同スキーマセットと異スキーマセットの比較

次に，異なるスキーマのデータセットに対して，提案手法を適用した結果を示す．マッチング判定結果を表 3 に示す．

表 3 画像特徴量による学習結果（異スキーマの場合）(%)

	Precision	Recall	F-Measure
TRUE	83.6	69.8	76.1
FALSE	99.5	99.8	99.7
Weighted Avg.	99.3	99.3	99.3

本結果から，TRUE の適合率は同スキーマの場合に比べて 4 ポイント程度の下落に留まったが，再現率は 13 ポイントの下落となっていることが分かる．具体的な失敗例としては，歌手名，グループ名，アルバム名などにおいて前節でも述べた現象（基準点の類似度が低くなり FALSE 判定）がより多くの項目で確認された．これは異スキーマの場合はプロパティ名も完全には一致しないためである．同義のリンクであっても Freebase は独自のプロパティを定義しており，例えば DBpedia の `http://ja.dbpedia.org/property/birthName` と Freebase の `http://rdf.freebase.com/ns/type.object.name` であれば，類似性はあるものの完全一致はしないため，同義語の周辺リンク・ノードの類似度が下がってしまい，FALSE と判定され易くなっている．今回，実験のために収集した範囲では，DBpedia と Freebase 間で共通しているプロパティは `http://www.w3.org/2000/01/rdf-schema#label` のみであった．更に，プロパティの種類は DBpedia のほうが多く，必ずしも対応するプロパティが Freebase 側にはなかった．こうした違いから十分に discrete なプロパティ組み合わせを学習できなかったものと思われる．今後，この問題に対して再現率を向上させるために，上記に該当する事例をより多く集め，モデル作成に十分なデータセットを構築することが必要だと思われる．

また，リソースの比較に関して，本実験では式 (3) で表したように Literal の比較と同様，リソース名の String Similarity と Semantic Similarity を取得している．しかし，スキーマセットによってはリソース名が ID（英数字の文字列）などになっており，同スキーマ間であれば何らかの規則性を String Similarity で捉える場合もあるが，異スキーマで比べた場合はいずれの Similarity も有効ではない場合がある．今回対象とした Freebase はそうしたスキーマセットであり，リソース名が m.01ksqk などの文字列となっている．この問題に対し，本手法では 1 点の特徴量に頼ることなく，周辺の特徴量（の差分）に重要度を分散することで解決を試みている（次節参照）．その結果，DBpedia との比較においても，同一ペアを F 値 76.1%にて判定可能であることを確かめた．今後，リソース名同士の比較が有効でない他のスキーマセット間でも比較実験を進めたい．一方，リソース名やその説明が `rdfs:label` や `rdfs:comment` などのプロパティを介して記述されていることが事前に分かっている場合は，それをリソース名として用いる，またはブロック画像を作成する際に，現状ではアルファベット順に 3 つ選ぶところを一定の箇所に必ず組み入れるようにすることで，より精度よく判別できるようになると思われる．

4·4 類似度の勾配による特徴量と絶対値による特徴量の比較

更に，特徴量として隣接する点からの差分（勾配）を用いた場合と絶対値を用いた場合との比較実験を行った．ここでは勾配を取ることで絶対値よりも汎用的な特徴量となっているかを確認するため，類似度の相対的なレベルが異なる，同スキーマのデータセットと異スキーマのデータセットを結合した約 3 万ペアに対して実験を行った．結果としては，勾配のほうが TRUE の適合率が僅かながら向上したが (84.7% → 85.5%)，TRUE の再現率 (約 77%) や FALSE の適合率，再現率ともにほぼ差がなかった．

一方で，両特徴量における説明変数の重要度（Random Forest における特徴量加工による重要度，Mean Decrease Accuracy）に関しても比較を行った（図 6）．結果として，いずれも基準点（対象 Literal 値の類似度）の重要度が他に比べて高いことが確認できるが，絶対値による特徴量に比べ，勾配による特徴量のほうがその重要度は下がっており，周辺のノード・リンクの重要度が上がっていることが確認できた．そこで絶対値と勾配それぞれの特徴量の標準偏差を測ってみたところ，それぞれ 0.36，0.54 となっており勾配のほうが大きいことが分かった．これは直接的には，勾配にしたことでマイナスの値が現れたことなどに起因すると思われるが，その結果，特徴量が変化に富むことで学習器が特徴を捉えやすくなったと推察される．これにより，基準点 1 点への依存が減ったものと思われる．上記の結果から，勾配による特徴量は絶対値と同等の精度を実現しながら，かつ特定の説明変数への依存を減らし，学習データの欠損や変化に強くなっていると言える．

5. まとめと今後の課題

本論文では，LOD に一定の割合で存在する Literal 値に URI を与え，複数の LOD を横断して検索可能とするため，LOD における Literal 値の同一性を判定するという問題を提案した．また，LOD から画像的な特徴量を抽出する手法を新規に考案した．実験では，DBpedia と Freebase の J(-)POP カテゴリから Literal 値を含むトリプルを抽出し，アンサンブル学習によってペア毎に同一性を判別したところ， マッチングペアに関して F 値 76–85%で判定可能であることを確かめた．

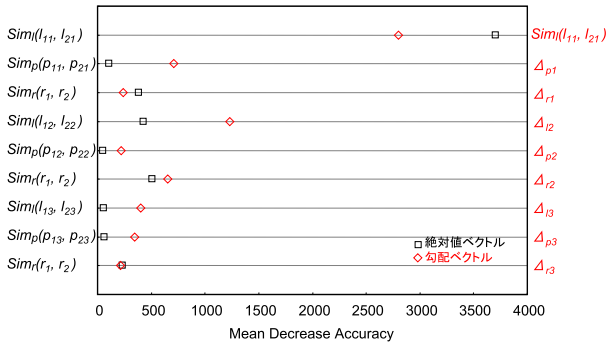

図 6 説明変数の重要度の比較

本手法は，原理的には分野（今回は JPOP カテゴリ）に依存しないものであるが，今後は他分野への適用を検討したい．また，構築した分類器をモジュール化し，動的にマッチングを判定できるサービスとして公開したい．こうした取り組みにより，LOD 上のリンクを辿った分野横断検索がより容易になることを期待している．

◇ 参 考 文 献 ◇

[Apache 14] Apache Solr, http://lucene.apache.org/solr/ (2014)

[Bizer 09] C. Bizer, T. Heath, and T. Berners-Lee: Linked Data - The Story So Far, International Journal on Semantic Web and Information Systems (IJSWIS), Vol. 5, No. 3, pp. 1-22 (2009)

[Breiman 01] L. Breiman: Random Forests, Machine Learning, Vol. 45, No. 1, pp. 5-32 (2001)

[Castano 11] S. Castano, A. Ferrara, S. Montanelli, and G. Varese: Ontology and Instance Matching, Knowledge-Driven Multimedia Information Extraction and Ontology Evolution, pp. 167-195 (2011)

[Dai 07] W. Dai, Q. Yang, G. Xue, and Y. Yu: Boosting for transfer learning, Proc. of 24th International Conference on Machine Learning (ICML), pp. 193-200 (2007)

[DBpedia 14] National Institute of Informatics: DBpedia Japanese, http://ja.dbpedia.org/sparql (2014)

[Google 14] Google Refine (in transition over to Open Refine), https://github.com/OpenRefine (2014)

[Gunaratna 13] K. Gunaratna, K. Thirunarayan, P. Jain, A. Sheth, and S. Wijeratne: A Statistical and Schema Independent Approach to Identify Equivalent Properties on Linked Data, Proc. of 9th International Conference on Semantic Systems (I-SEMANTICS), pp. 33-40 (2013)

[Hogan 06] A. Hogan, A. Harth, and S. Decker: ReConRank: A Scalable Ranking Method for Semantic Web Data with Context, Proc. of 2nd International Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS), (2006)

[Hu 10] W. Hu, J.Chen, G. Cheng, and Y. Qu: ObjectCoref & Falcon-AO: Results for OAEI2010, Proc. of 5th International Workshop on Ontology Matching, (2010)

[Isele 11] R. Isele and C. Bizer: Learning linkage rules using genetic programming, Proc. of 6th International Workshop on Ontology Matching (OM), (2011)

[Li 09] J. Li, J. Tang, Y. Li, and Q. Luo: Rimom: A dynamic multistrategy ontology alignment framework, IEEE Transactions on Knowledge and Data Engineering, Vol. 21, No. 8, pp. 1218-1232 (2009)

[Kuromoji 14] atilika: “kuromoji”, http://www.atilika.com/ja/products/kuromoji.html (2014)

[Lowe 99] D. G. Lowe: Object recognition from local scale-invariant features, Proc. of 7th International Conference on Computer Vision (ICCV), Vol. 2, pp. 1150-1157 (1999)

[Maali 11] F. Maali, R. Cyganiak, and V. Peristeras: Re-using Cool URIs: Entity Reconciliation Against LOD Hubs, Proc of 4th Linked Data on the Web Workshop (LDOW 2011), (2011)

[Niu 12] X. Niu, S. Rong, H. Wang, and Y. Yu: An Effective Rule Miner for Instance Matching in a Web of Data, Proc. of 21st ACM international conference on Information and knowledge management (CIKM), pp. 1085-1094 (2012)

[OpenLink 14] OpenLink Software: OpenSearch, http://lod.openlinksw.com/sparql/ (2014)

[Raimond 08] Y. Raimond, C. Sutton, and M. Sandler: Automatic Interlinking of Music Datasets on the Semantic Web, Proc. of Linked Data on the Web (LDOW), (2008)

[Rong 12] S. Rong, X. Niu, E. W. Xiang, H. Wang, Q. Yang, and Y. Yu: A Machine Learning Approach for Instance Matching Based on Similarity Metrics, Proc. of 11th International Semantic Web Conference (ISWC), pp. 460-475 (2012)

[Sindice 14] Sindice - The Semantic Web Index, http://sindice.com/, (2014)

[Volz 09a] J. Volz, C. Bizer, M. Gaedke, and G. Kobilarov: Silk - A link discovery framework for the web of data, Proc. of 2nd Linked Data on the Web Workshop (LDOW), (2009)

[Volz 09b] J. Volz, C. Bizez, M. Gaedke, and G. Kobilarov: Discovering and maintaining links on the web of data, Proc. of 8th International Semantic Web Conference (ISWC), pp. 650-665 (2009)

[Wielemaker 08] J. Wielemaker, M. Hildebrand, J. Ossenbruggen, and G. Schreiber: Thesaurus-based search in large heterogeneous collections, Proc. of 7th International Semantic Web Conference (ISWC), pp. 695-708 (2008)

[芦川 14] 芦川 将之，川村 隆浩，大須賀 昭彦: マイクロタスク型における精度向上手法を導入した専用クラウドソーシングプラットフォームの構築, 人工知能学会論文誌, Vol. 29, No. 6. pp. 503-515 (2014)

[石井 12] 石井 愛: 検索技術による企業内外データの仮想統合, UNISYS TECHNOLOGY REVIEW, Vol. 111, pp. 79-89 (2012)

[酒井 13] 酒井 一晃: みんなビックデータビックデータって言ってるけど名寄せとかどうしてんの?, http://www.slideshare.net/send_/namebased-aggregation, (2013)

〔担当委員：古崎 晃司〕

2014 年 8 月 8 日 受理

著　者　紹　介

川村　隆浩（正会員）

1994 年早稲田大学大学院 理工学研究科 電気工学専攻 修士課程了．同年 (株) 東芝入社．現在，同社 研究開発センター 主任研究員．2001-2002 年 米国カーネギ・メロン大学 ロボット工学研究所 客員研究員 兼任．2003 年より電気通信大学大学院 情報システム学研究科 客員准教授 兼任．2007 年より大阪大学大学院 工学研究科 非常勤講師 兼任．工学博士 (早稲田大学)．主としてセマンティック Web，エージェント技術の研究・開発に従事．2012-2013 年 人工知能学会理事．情報処理学会会員．

長野　伸一（正会員）

1999 年大阪大学大学院 基礎工学研究科 博士後期課程了．同年 (株) 東芝入社．現在，同社研究開発センター 主任研究員．2004-2006 年 国立情報学研究所 特任講師 兼任．2010-2012 年 法政大学 兼任講師 兼任．博士 (工学)．主として Linked Data，セマンティック Web の研究・開発に従事．2011, 2013 年 Linked Open Data チャレンジ Japan 実行副委員長．2014 年 人工知能学会理事．電子情報通信学会，情報処理学会 各会員．

大須賀　昭彦（正会員）

1981 年上智大学 理工学部 数学科卒．同年 (株) 東芝入社．同社 研究開発センター，ソフトウェア技術センター等に所属．1985-1989 年 (財) 新世代コンピュータ技術開発機構 (ICOT) 出向．2007 年より電気通信大学 大学院情報システム学研究科 教授．2012 年より国立情報学研究所 客員教授 兼任．工学博士 (早稲田大学)．主としてソフトウェアのためのフォーマルメソッド，エージェント技術の研究に従事．1986 年度情報処理学会論文賞受賞．IEEE Computer Society Japan Chapter Chair，人工知能学会理事，日本ソフトウェア科学会理事を歴任．電子情報通信学会，情報処理学会，日本ソフトウェア科学会，IEEE Computer Society 各会員．