Vol.2014-NL-217 No.9
2014/7/4

近似頻度計測手法を用いた大規模データからの関係獲得

高瀬 翔[1,a)]　岡崎 直観[1,2,b)]　乾 健太郎[1,3,c)]

概要：大規模な Web データからエンティティペアやパタンのクラスタリングによって意味関係を抽出する教師なし関係抽出タスクにおいて，パタンやエンティティペアの間の類似度計算をスケーラブルに行う事は非常に重要な問題である．本論文では，近似頻度計算による省メモリな素性作成手法と次元圧縮による類似度計算時間の高速化を実現する手法を提案する．実験において，近似計算でも精度が落ちない事，また，計算時間の高速化の効果を示す．

1. はじめに

エンティティ間の意味的関係は質問応答や推論，情報抽出など自然言語処理における様々な応用タスクに必須の知識である [20], [25]．このため，意味的関係のインスタンス（例えば AuthorOf という意味的関係については AuthorOf⟨フランツ・カフカ，変身⟩）を網羅的に収集し，知識ベースを構築しておく事は非常に重要な課題である．コーパスからインスタンスを収集するために，エンティティ間の意味的関係を認識するためのパタン（例えば AuthorOf という意味的関係については “X が Y を書く”）をあらかじめ獲得しておかなければならない．

従来は，特定の意味的関係に属するパタンやインスタンスを収集する研究が盛んに行われて来たが [4], [19], [22]，近年では，より幅広い知識を得るため，Open Information Extraction (Open IE) が注目を集めている [2]．Open IE とは，対象となる意味的関係をあらかじめ定めず，大規模なコーパスから，あらゆるインスタンスやパタンを自動で獲得する事を目的としたタスクである [2], [5], [11], [26]．Open IE では意味的関係の抽出と，パタンがどの意味的関係に属するかを知る事が必要であり，このタスクは教師なし関係抽出と呼ばれる [9], [23], [24].

教師なし関係抽出は一般的に，同一の意味関係を表すパタン毎のクラスタを構築する事で行う [9], [16], [17], [23], [24], [27]．すなわち，各クラスタはそのクラスタの意味関係に対応したパタンのみを含有する．例えば，“X が Y を書く”，“X が Y の著者である”，“X は Y に位置する” という三つのパタンについて考えてみよう．これらを同一の意味関係毎にまとめたとすると，“X が Y を書く” と “X が Y の著者である” というパタンは AuthorOf という意味関係のクラスタとなり，“X は Y に位置する” は LocatedIn という意味関係のクラスタとなる．このようなクラスタを得るためにはパタン間の意味的類似度を知る必要があり，意味的類似度を精度良く得る事ができれば，クラスタリングの精度もそれに応じて向上する．従って，パタン間の意味的類似度の計算は教師なし関係抽出において中核的な問題である．

パタン間の意味的類似度計算には，主要な問題が二つある．一つ目は，パタンの意味をどのように表現するか，という問題である，既存の研究には，パタンの素性空間を定義し，エンティティペアとの共起頻度や自己相互情報量（PMI）など共起情報でパタンの意味を表現するもの [18] や，主成分分析（PCA）や Latent Dirichlet Allocation（LDA）などを利用し，固定次元のベクトルによってパタンの意味を表現するもの [21], [27] などがある．しかしながら，各表現手法について，大規模なコーパス上での有効性は十分に調査されていない．

二つ目は，大規模なデータに対しスケーラブルな手法でなければならないという問題である．教師なし関係抽出の質を向上させる，すなわち，大量のパタンとその表す意味関係を知るためには非常に大規模なデータを利用する事となる．このため，計算時間やメモリを大幅に消費するような，複雑で非効率的なアルゴリズムは採用できない．ここで，ランダムサンプリングを行いデータ量を削減するという解決法も考えられるが，データ全体に対する処理が不可能であれば，ランダムサンプリングにより結果の質がどの

[1] 東北大学
Tohoku University
[2] 科学技術振興機構・戦略的創造研究推進事業「さきがけ」
PRESTO, Japan Sciencce and Technology Agency (JST)
[3] 科学技術振興機構・戦略的創造研究推進事業「CREST」
CREST, Japan Sciencce and Technology Agency (JST)
[a)] takase(at)ecei.tohoku.ac.jp
[b)] okazaki(at)ecei.tohoku.ac.jp
[c)] inui(at)ecei.tohoku.ac.jp

程度低下するのか評価する事が難しいため，解決法として望ましくない．

本論文では，数十億という非常に大規模なデータに対しスケーラブルであり，かつパタン間の意味的類似度を精度良く計算できる手法を調査する．具体的には，近似頻度計算と次元圧縮を利用した教師なし関係抽出システムを設計し，その性能を評価する．本研究の貢献は以下のとおりである．

- 大規模なデータに対し，スケーラブルであり，実用的な教師なし関係抽出システムの構築．
- 近似計算手法を用いても，正確に計算した場合と同等の精度が達成可能である事を実験を通して示す．
- パタンの意味について，いくつかの表現手法を比較し，意味的類似度を測るために適切な手法を探る．

本論文の構成は以下のとおりである．2 節では教師なし関係抽出システム構築における問題点について述べ，さらに，近似頻度計算と次元圧縮を用いたシステムの提案を行う．大規模な日本語コーパスにこのシステムを適用する事により，その有用性を 3 節において示す．その後，4 節で関連研究について述べ，5 節において，結論と今後の展望について述べる．

2. システムの構築

1 節で記したように，教師なし関係抽出においては，パタンの意味を適切に表現することが重要である．分布仮説に基づくと [6]，パタンと共起するエンティティペアの分布によって，パタンの意味をモデル化することができると考えられる．例として，“X が Y を書く” というパタンの意味は，コーパス中で変数部分（X，Y）に出現するエンティティペアの分布によって表される．この分布のベクトル表現を用いる事で，パタン間の意味的類似度を計算する事が可能である．例えば，“X が Y を書く” と “X が Y の著者である” という二つのパタンについて，共起するエンティティペアの分布が似ていた場合，互いに似た意味を持つと推測できる．

パタンの意味を表現するベクトルの構築に関して，様々な取り組みがなされてきた [10], [21], [23], [27]．最も単純な手法としては，パタンと共起するエンティティペアおよびその共起頻度によってパタンの意味を表す事である．しかしながら，ほとんどのパタンと共起するエンティティペアが存在するような場合，このエンティティペアを介して多くのパタン間の意味的類似度が高くなってしまうため，共起頻度をそのまま利用するのは適切ではない．この問題を解決するために，頻度よりも洗練した尺度である PMI を使うなどがされている [10]．さらに，高次元で疎なベクトルを圧縮するために，PCA のような次元圧縮によって最終的なパタンの意味を表すベクトルを得る手法もある [21], [27]．

一方で，上記のような処理を大規模なデータに適用する事はスケーラビリティの点で難しいと考えられる．例として，150 億の文から 110 万のエンティティペアと 70 万のパタンを得たような状況について考えてみると，パタンとエンティティ間の組み合わせは最大で 7,700 億個である．このように，パタンとエンティティ間の組み合わせは大量に存在するため，仮にベクトル空間が疎であったとしても，共起頻度をカウントするためには膨大な記憶領域が必要となる．

本研究では，大規模なデータからパタンの意味を表現するベクトルを得るために，二つの手法を考える．一つ目は，近似頻度カウントの利用により，重要度の低いと考えられる要素を除去しつつパタンのベクトルを構築する手法であり，もう一方は，単語の意味を表す固定次元のベクトルを利用する事により，小さな固定次元の空間でパタンのベクトルを構築する手法である．言い換えれば，前者の手法はパタンのベクトルの要素を計算する際のメモリ使用量を抑える手法であり，後者はベクトル空間をあらかじめ圧縮しておく手法である．

図 1 は本論文で提案する教師なし関係抽出システムの概要を示したものである．まず，エンティティペアとパタンからなる triple をコーパス中から収集する．このとき，何らかの意味関係を表す triple を得られるよう，コーパス中で出現頻度の高いエンティティペアとパタンから構成されるものに限定する．次にエンティティペアとパタン間の共起統計を計算し，パタンのベクトルを得る．得られたパタンベクトルは高次元なため，後々の計算時間を短縮するために，PCA を用いて次元圧縮する．上記に加え，大規模データをスケーラブルに扱うための二つの手法について述べる．

2.1 triple の抽出

本研究では，エンティティペアと，その二つのエンティティを結ぶパタンとの組み合わせを triple と定義する．意味関係を表す tiple を抽出するために，まず意味関係を表すと考えられるエンティティとパタンを収集し，その後，triple の抽出を行う．

2.1.1 エンティティの抽出

本研究では，名詞に加え，連続した名詞列および “名詞の名詞” という形で記述された名詞句のうち，複合名詞のように一個の名詞として認められるものをエンティティと見なす．このため，エンティティの中には “道の駅” や “村上春樹作品” のように二語以上から成るものが存在する．これらを認識するため，単純な統計的手法 [14] を用いて名詞連続 “$w_i w_j$” および “w_i の w_j” がエンティティとして見なせるかを判定する．すなわち，式 1 を計算する事で，名詞列のエンティティらしさを得る．

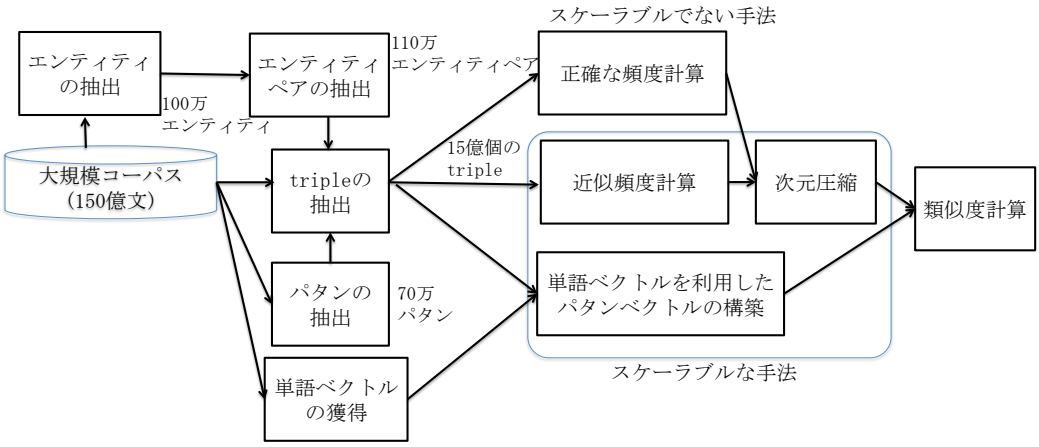

図 1 教師なし関係抽出システムの概要図

$$\mathrm{score}(w_i, w_j) = \mathrm{cor}(w_i, w_j) * \mathrm{dis}(w_i, w_j), \quad (1)$$

$$\mathrm{cor}(w_i, w_j) = \log \frac{f(w_i, w_j) - \delta}{f(w_i) \times f(w_j)}, \quad (2)$$

$$\mathrm{dis}(w_i, w_j) = \frac{f(w_i, w_j)}{f(w_i, w_j)+1} \frac{\min\{f(w_i), f(w_j)\}}{\min\{f(w_i), f(w_j)\}+1}. \quad (3)$$

ここで，$f(w_i)$ は名詞 w_i の頻度，$f(w_i, w_j)$ は名詞連続 $w_i w_j$ の出現頻度であり，δ は頻度の少ない名詞列を除去するための定数である．すなわち，$\mathrm{cor}(w_i, w_j)$ は名詞列 $w_i w_j$ の結合度を表す．この値は名詞列 $w_i w_j$ が偶然ではないと考えられるほど出現している場合に大きくなるが，$f(w_i)$ や $f(w_j)$ が小さな場合に，それに影響されて大きくなってしまうという性質もある．これを防ぐため，$\mathrm{dis}(w_i, w_j)$ という項を導入し，$f(w_i)$ や $f(w_j)$ が小さな場合に値が大きくなる事を抑える．score を計算した結果，しきい値以上のものを一個の名詞として扱うものとする．

三語以上の名詞連続からなる名詞や “名詞の名詞” 形式のものも扱えるようにするため，上記の操作を名詞連続 $w_i w_j$ について，しきい値を下げつつ 3 回行った後，w_i の w_j 形式についての名詞を獲得し [*1]，最終的にエンティティと見なす文字列を得る [*2]．これにより，例えば，最初に “村上春樹” を得たとすると，2 周目には “村上春樹作品” を得る事ができる．エンティティとして扱う名詞句が定まったら，コーパス中での頻度をカウントし，頻度 1,000 以上のものを抽出する．

2.1.2 エンティティペアの抽出

頻度の高いエンティティを抽出した後，パタンで結ばれた際に意味的関係を持つと考えられるエンティティペアを抽出する．本研究では，文中での共起頻度が 5,000 回以上のエンティティペアを抽出する．この抽出したエンティティペアの集合を E と呼ぶ事とする．

2.1.3 パタンの抽出

本研究では，既存研究にならい [1], [11], [26]，係り受け木上でのエンティティ間の最短パスをパタンとして扱う．なお，“X の Y” のような意味的に曖昧なパタンを除くため，述語を含むパスのみをパタンとする．さらに，エンティティの部分は変数（X，Y）に置き換える事とする．例えば図 2 に書かれた文について考えてみる．この図では文は文節に分けられており，文節間の矢印は係り受け関係を表すものとする．この文に含まれるエンティティは “カフカ”，“変身”，“ドイツ語” の三つであるので，パタンとしては，“X は → 書いた ←Y を”，“X は → 書いた ←Y で”，“X で → 書いた ←Y を” の三種類となる．コーパス中でのパタンの頻度を数え，1,500 回以上出現しているものを抽出する．この抽出したパタンを P と呼ぶ事とする．

[*1] このとき，式 1 の $w_i w_j$ を w_i の w_j と読み替える
[*2] しきい値は 10，5，0 と下げていく．なお，“名詞の名詞” に関してはしきい値を 10 とする．

2.2 パタンベクトルの構築

パタンの意味を表すベクトル（パタンベクトル）を，パタンと共起するエンティティペアの分布から構築する．まず，コーパス全体から，triple (p, e)，$p \in P$，$e \in E$ を抽出する．例えば，“フランツ・カフカが変身を書いた” という文からは，以下のような triple が得られる．

$p =$ X は → 書いた ← Y を,
$e =$ ⟨“カフカ”, “変身”⟩,

本研究では，共起情報を二つの統計的手法，すなわち，共起頻度（Freq）と PMI（Pmi）で表現する．Freq では，p のベクトルは各要素がエンティティペア $e \in E$ との共起頻度 $f(p, e)$ からなる．PMI では，共起頻度を以下の式により洗練する．

$$\mathrm{PMI}(p, e) = \log \frac{\frac{f(p,e)}{M}}{\frac{\sum_{i \in P} f(i,e)}{M} \frac{\sum_{j \in E} f(p,j)}{M}} \times \mathrm{dis}(p, e). \quad (4)$$

ここで，$f(p, e)$ はパタン p とエンティティペア e 間の共起頻度を表し，$M = \sum_i^P \sum_j^E f(i, j)$ とする．$\mathrm{dis}(p, e)$ は式 3 と同様のものである．なお，しきい値として 0 を設ける，すなわち PMI では $\mathrm{PMI}(p, e) < 0$ のときにエンティティペ

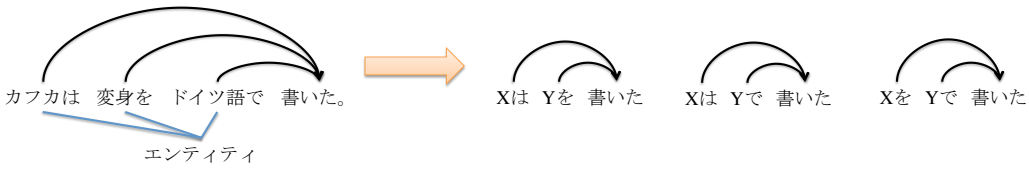

図 2 係り受け解析済みの文と抽出されるパタンの例

ア e とパタン p との値を 0 とする．

2.3 パタンベクトルの次元圧縮

2.2 節で定義したベクトルは，全エンティティペアを別々の次元として扱っているため，非常に高次元かつ疎になっており，パタンの意味を表現するのに適切でない可能性がある．例えばエンティティペア（“フランツ・カフカ”，“変身”）と（“カフカ”，“変身”）について，“フランツ・カフカ” と “カフカ” が同一の人物であるにも関わらず，別々の次元として扱っている．さらに，疎なベクトル間の類似度を計算しようとした場合，連想配列を用いなければならなくなってしまう．

高次元で疎なベクトルを低次元かつ密なベクトルに写像するため，主成分分析（PCA）を用いる．PCA とは，行列の主成分および主成分スコアを求める統計的な手続きであり，特異値分解（SVD）と密接な関係がある．$m \times n$ の行列 $\mathbf{A}$ に対する SVD は以下の式で表される．

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^t. \tag{5}$$

ここで，$\mathbf{U}$ は $m \times m$ の直行行列，$\mathbf{V}$ は $n \times n$ の直行行列であり，$\mathbf{\Sigma}$ は特異値を対角成分とする $m \times n$ の対角行列である．$\mathbf{V}$ の各列は $\mathbf{A}$ の主成分に対応し，$\mathbf{U\Sigma}$ の各列は $\mathbf{A}$ の主成分スコアに対応する．すなわち，特異値分解を行う事で主成分分析の結果を得る事ができる．

パタンベクトルの次元を r まで圧縮する場合，求めたいものは行列 $\mathbf{A}$ の上位 r 個の特異値に対応する主成分スコアである．しかしながら，パタンとエンティティペアとの共起情報を要素に持つ巨大な行列全体に対し，特異値分解を行う事は膨大な計算量となってしまう．この問題を解決するため，Halko ら [8] により提案された乱択アルゴリズムを利用する．

このアルゴリズムの目的は $\mathbf{A}$ の行を圧縮し，$r \times n$ の行列 $\mathbf{B}$ を得る事である．まず始めに，各成分が平均 0，分散 1 のガウス分布からサンプルされた $n \times r$ のガウシアン行列 $\mathbf{\Omega}$ を作成する．次に，$\mathbf{Y} = \mathbf{A\Omega}$ により，$m \times r$ の行列 $\mathbf{Y}$ を得る．さらに，$\mathbf{Y}$ の各列を正規直行化する事により，$m \times r$ の行列 $\mathbf{Q}$ に変形する．ここで，$\mathbf{QQ}^t\mathbf{A} \approx \mathbf{A}$ が成り立つ．最終的に，$\mathbf{B} = \mathbf{Q}^t\mathbf{A}$ より $\mathbf{B}$ を得る，すなわち，$\mathbf{Q}^t$ により，行列 $\mathbf{A}$ の各行を圧縮する．

$\mathbf{B}$ に SVD を行う事により，上位 r 個の特異値に対応する主成分スコアを得る．このとき，$r \ll m$ であるため，元々の行列 $\mathbf{A}$ に対して SVD を行うよりも，遥かに計算量は少ない．本研究では，Halko ら [8] の手法を実装したものとして，redsvd[*3] を用いる．得られた r 次元の主成分スコアをパタンの意味を表すベクトルとして用いる．

Algorithm 1 各パタンに対する Space Saving

Input: N: 各パタンの有するカウンタのサイズ
Input: D: トリプル (p, e) の集合
Output: $c_{p,e}$: 各パタン p に対するカウンタ

for all $(p, e) \in D$ **do**
  if T_p がない場合 **then**
    $T_p \leftarrow \emptyset$
  end if
  if $e \in T_p$ **then**
    $c_{p,e} \leftarrow c_{p,e} + 1$
  else if $|T_p| < N$ **then**
    $T_p \leftarrow T \cup \{e\}$
    $c_{p,e} \leftarrow 1$
  else
    $i \leftarrow \mathrm{argmin}_{i \in T_p} c_{p,i}$
    $c_{p,e} \leftarrow c_{p,i} + 1$
    $T_p \leftarrow T \cup \{e\} \setminus \{i\}$
  end if
end for

2.4 大規模データに対するスケーラビリティの向上

先に記したように，巨大なデータから共起頻度を正確にカウントする事は非効率的であり，難しい問題でもある．これを解決するため，本研究では二つの手法，すなわち，近似頻度計算と単語の意味の固定次元ベクトル表現を用いる．

2.4.1 近似頻度計算

エンティティペアとパタンベクトルとの共起情報について，パタンベクトル間の類似度計算への寄与が大きい要素は少ないと考えられるため，パタンベクトル作成のために共起頻度を正確に得る必要はないと考えられる．すなわち，各パタンについて共起頻度の上位 k 個のエンティティペアだけを取得し，それ以外は無視してしまっても類似度計算に影響はないと考えられる．このような，上位 k 個の頻出アイテムを発見するタスクについては近似カウントアルゴリズムが有用である．

本研究では，頻度上位 k 個のアイテムを得るために，最

[*3] https://code.google.com/p/redsvd/wiki

も有効なアルゴリズムである，*Space Saving* [12] を利用する．Algorithm 1 は Space Saving アルゴリズムを用いて共起頻度カウントを行う際の概要を示したものである．Space Saving アルゴリズムは，各パタン p について，共起頻度をカウントするために最大 N 個のカウンタを有する，すなわち，N 個のアイテムの頻度を保持する．

このアルゴリズムでは，まず，パタン p とエンティティペア e からなる triple (p, e) について，共起頻度を集計中であるか否か，言い換えれば，パタン p のカウンタにエンティティペア e が含まれているかどうかをチェックする．もし共起頻度を集計中である場合，（Line 5）共起頻度に 1 を加算する（Line 6）．共起頻度を集計中でなく，さらにパタン p のカウンタに含まれるアイテム数が N よりも少ない場合（Line 7），エンティティペア e を共起頻度 1 としてカウンタに記録する（Line 8 および Line 9）．さらに，もし共起頻度を集計中でなく，かつカウンタが既に N 個のアイテムを保持していた場合，最も小さな値 $c_{p,i}$ を持つアイテムを除去し，新たにエンティティペア e を頻度 $c_{p,i}+1$ としてカウンタに記録する（Line 11-13）．

このアルゴリズムは，アイテムの頻度の誤差について，カウンタのサイズに依存する値以内に収まり，さらに，頻出するアイテムであるほど正確な値を保持するという性質がある．このアルゴリズムでは，k の値よりもカウンタのサイズ N を十分大きく取っていれば，上位 k 個の頻出アイテムとその頻度をほぼ正確に得る事ができる．この手法により，各パタンについて上位 k 個の頻出エンティティペアとその頻度 $f(p, e)$ を獲得し，残りのエンティティペアとの共起頻度は 0 として，パタンベクトルを構築する．

2.4.2 単語ベクトルを用いたパタンベクトルの構築

近年，多くの NLP 研究者が，単語の意味を固定次元のベクトルで表現する手法を提案している [3], [14]．特に Mikolov ら [14] の手法を実装した word2vec[*4] は，高品質な単語の意味ベクトルを構築でき，さらに作成したベクトルの足し引きで意味の計算が可能という特性から，多くの研究者の注目を集めている．

本研究での目的は，パタンと共起するエンティティペアの分布からパタンの意味を表現するベクトルを構築する事である．そこで，大規模コーパスから word2vec によって学習した，低次元の単語ベクトルからパタンベクトルを構築する，すなわち，エンティティのベクトルの分布から，パタンベクトルを構築する事を考える．パタン p について，エンティティのベクトルを利用したベクトル表現を以下の式で得る．

$$\boldsymbol{p} = \sum_{e \in E} f(p, e) \begin{bmatrix} \boldsymbol{v}_{e_0} \\ \boldsymbol{v}_{e_1} \end{bmatrix}. \quad (6)$$

ここで，$\boldsymbol{v}_{e_0}$ はエンティティペア e に含まれるエンティティ e_0 のベクトルであり，$\boldsymbol{v}_{e_1}$ は e に含まれるもう一方のエンティティ e_1 のベクトルである．

3. 実験

3.1 データ

実験のために，日本語 Web ページをクローリングし，150 億の文を得た．スパム広告や日本語以外の文を除去するため，文が長過ぎない／短過ぎないか，ひらがなやカタカナ，漢字を含むか，記号は多すぎないかなどのフィルタを適用した．このフィルタリングの結果，63 億の文を獲得し，CaboCha[*5] によって係り受け構造を解析した結果をコーパスとして利用する事とした．triple 抽出前の下処理において，100 万種類のエンティティ，110 万種類のエンティティペア，70 万種類のパタンを獲得し，最終的に，15 億個の triple を獲得した．

さらに Wikipedia の情報を元に，同一の意味関係を表すと考えられるパタンペアを抽出し，人手で正否を付与する事で評価用データを作成した．まず 70 万種類のパタンの中から，特定の意味関係を表すパタンが得られるよう，Wikipedia 上でのドメインを病気，著者，建造物に絞り，各ドメインのページで頻度の高いパタンを抽出した．その後，各パタンについて，Wikipedia 中で共起するエンティティペアを多く共有するものを，同一の意味関係を表す可能性があるパタンペアとして抽出した．このパタンペアの数は 4,531 個であった．さらに，4 人の評価者にパタンペアが同一の意味関係を表しているか否かの判定を依頼した．4,531 個のパタンペアについて，4 人中 2 人以上の評価者が正否を付与しており，確実に同一の意味関係を表すと考えられるペアのみを正解とするため，もし 2 人以上が同一の意味関係と見なしたときのみ，そのパタンペアを正解として扱う事とした．なお，ランダムに 90 ペアをサンプリングしたときの，評価者二人での一致率（カッパ値）の平均値は 0.63 であった．最終的に同一の意味関係を表すパタンペアとして 720 個のペアを得た．実験では，各手法を 4,531 個のパタンペアに適用し，類似度がしきい値以上のペアについて，同一の意味関係を表すペアであるかを判定する事で性能を評価した．

3.2 実験設定

スケーラビリティの向上のために導入した手法によって得られるパタンベクトルの質について，パタン間の類似度計算によって評価した．具体的には，精度への影響，メモリ使用量，計算時間について調べた．実験では，以下の各手法によって構築されたパタンベクトルについて，コサイン類似度を計算した．

[*4] https://code.google.com/p/word2vec/

[*5] https://code.google.com/p/cabocha/

正確な頻度計算 (baseline)

256GB のメモリを有する計算機でエンティティペアとパタンの共起頻度を正確に計算し，パタンベクトルとした（EXACT-FREQ）．また，この共起頻度の計算結果を元に，式 4 から得られる PMI を計算した（EXACT-PMI）．

正確な頻度計算 + PCA

EXACT-FREQ と EXACT-PMI を PCA を用いて，固定次元のベクトル EXACT-FREQ+PCA および EXACT-PMI+PCA に変換した．次元数については，256，512，1,024，2,048 次元を比較し，精度が最も良かった 1,024 次元を採用した [*6]．

近似頻度計算

2.4.1 節で述べた近似頻度計算アルゴリズムによって共起頻度の計算を行った（APPROX-FREQ）．カウンタのサイズ N は 10,240 個とし，頻度上位 5,120 個のエンティティペアを各パタンベクトルの要素とする事とした．さらに，この近似頻度計算の結果に基づいて PMI を計算した（APPROX-PMI）．

近似頻度計算 + PCA

APPROX-FREQ と APPROX-PMI を PCA によって固定次元のベクトル APPROX-FREQ+PCA および APPROX-PMI+PCA に変換した．正確な頻度計算 + PCA と同様に，次元数は 1,024 とした．

正確な頻度計算 + word2vec

word2vec から得られるエンティティのベクトルを用いてパタンベクトルを構築した（EXACT-FREQ+WORD2VEC）．さらに，共起頻度の代わりに PMI を用いてエンティティベクトルの重み付けを行い（すなわち，6 における $f(p, e)$ をパタン p とエンティティペア e の PMI とした）パタンベクトルを構築した（EXACT-PMI+WORD2VEC）．word2vec でのエンティティおベクトル獲得には，コーパス中のすべてのエンティティ，すべての動詞，すべての形容詞を用いて行った．なお，AMD Opteron 6174 (12-core, 2.2GHz) の CPU を 4 つ搭載した計算機で，42 スレッド，窓幅 5 で word2vec を実行したところ，130 時間を要した．PCA と同様，パタンベクトルの次元数を 1,024 次元とするため，エンティティのベクトルは 512 次元とした [*7]．

3.3 各手法での精度

各手法について，しきい値を変化させながら適合率と再現率を測定し，結果を図 3 に示した．まず，正確な頻度計算と近似頻度計算について比較を行うと，EXACT-FREQ と APPROX-FREQ ではほとんど同じ結果であり．APPROX-PMI は EXACT-PMI と比べて精度が向上していることが分かる．すなわち，パタンとエンティティペアの共起頻度計算においては，近似計算手法で十分であり，さらに，近似頻度計算手法はパタンの意味を表現するのに適切でないエンティティペアを削除する効果もあると考えられる．

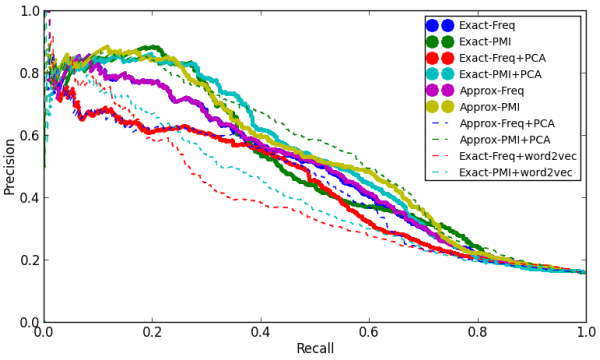

図 3 各手法での適合率と再現率

手法	非ゼロの次元数	メモリ使用量
Exact	380,497	3MB
Approx	10,240	80KB
word2vec	1,024	8KB

表 1 パタンベクトル獲得における，パタン一個に対するメモリの最大消費量

PCA を用いて次元圧縮した手法としていない手法とを比べると，PCA は常に精度を向上させるわけではないが，PMI で構築したベクトルに関しては精度を向上させる事ができ，特に，APPROX-PMI+PCA についてはほとんどの領域で最も良い結果を出力している．次元を圧縮する事により計算時間を削減できる事を考慮すると，各エンティティペアをそれぞれ別の次元として扱うよりは，有用な手法であると考えられる．

これに対して，パタンのベクトルを word2vec で得たエンティティベクトルを用いて構築した場合には，ほとんどの領域で最も悪い結果となってしまっている．これは，エンティティペアをペアのまま扱うのではなく，エンティティに分けた上でベクトルを構築してしまった事によるものと考えられる．この点については，誤り事例とともに，3.6 節で詳しく述べる．

図 3 から得られる結論としては，パタンの意味を表すベクトルの作成においては，近似頻度計算で十分であり，これに加えて PCA を用いる事で，余分な次元を圧縮し，低次元のパタンのベクトルを得る事が可能となる，という二点である．

3.4 メモリ消費量

各手法でのパタンベクトル作成の際に，ベクトル中での非ゼロ要素の最大数とそのときのメモリ使用量を表 1 に示した．すなわち，表 1 には，各手法における，パタン一つに対するメモリの最大消費量を示してある．ここで，メモリの消費量については，一つの次元が 8 バイトを有するも

[*6] 2,048 次元と 1,024 次元での結果はほぼ同一であり，類似度計算の短さから 1,024 次元を採用した．
[*7] word2vec を利用したパタンベクトルについては，次元数を変えても精度への影響はほとんどなかった．

手法	10k	100k	all (664k)
EXACT-PMI	55m	121hr	8,499hr
APPROX-PMI	38m	110hr	7,441hr
APPROX-PMI+PCA	4m	7hr	785hr

表 2 各手法における一スレッドでの類似度計算時間

のとして計算している [*8]．なお，表 1 には正確な頻度計算 **Exact**，近似頻度計算 **Approx**，word2vec で得られるベクトルを利用したパタンベクトル構築 **word2vec** の三つの手法についての結果を記してある．

表 1 から，**Approx** のメモリ使用量は **Exact** のおよそ 1/100 であり，記憶する要素を削減する事で，メモリ消費量を大幅に抑制している事が分かる．また，**word2vec** は **Approx** のさらに 1/10 の消費量に抑えており，本研究で利用した手法は，大規模データを十分スケーラブルに扱えるものであると考えられる．

3.5 計算時間

計算対象のパタンを 1 万個（10k），10 万個（100k），66 万 4 千個（664k）としたときの，EXACT-PMI，APPROX-PMI，APPROX-PMI+PCA の三つの手法についての類似度計算時間の結果を表 2 に示した．なお，計算時間は C++で実装されたプログラムを，CPU は AMD Opteron 6174（12-core, 2.2GHz），256GB のメモリを有する計算機で，1 スレッドで計算した結果である．なお，100k と 664k については 1 スレッドでは膨大な時間がかかってしまうため，パタンの組み合わせをそれぞれ 48 グループ，4096 グループに分割し，1 グループでの計算時間をグループ数倍したものを記載している．

APPROX-PMI は EXACT-PMI よりも非ゼロの素性数が少ないため，計算時間が減っているが，それでも 100k, 664k となると膨大な時間がかかってしまう．これに対し，APPROX-PMI+PCA は計算時間を他の手法の 1/10 まで減少させている．APPROX-PMI でも 664k のパタンに対しては 7,441 時間（約 1 年）かかる事を考慮すると，大規模データを扱うためには，次元数を小さくしておく事は必須であると言える．すなわち，実用的な時間で類似度計算を終えるためには，word2vec であらかじめ作成したエンティティのベクトルか，次元圧縮によって次元数を抑える必要がある．

3.6 誤り分析

表 3 に APPROX-PMI+PCA と EXACT-PMI+WORD2VEC の出力例として，“X を → 綴った → Y” と “X に ← 死去した → Y は” という二つのパタンに対する，上位 3 近傍のパタンを記した [*9]．この表において，出力したパタンのうち，対象のパタンと同一の関係を表していないものについては*を付与した．

EXACT-PMI+WORD2VEC は “X を → 綴った → Y” について “X で ← 書いた → Y を” という誤ったパタンを出力してしまっている．これは，“X で ← 書いた → Y を” と対象のパタンとで，X に当てはまるエンティティのタイプが同じ（すなわち，X には「本」や「作品」が入る）ためであると考えられる．本研究では，word2vec による，単語の意味を表現したベクトルからパタンベクトルを得る際に，X に入るエンティティのベクトルと Y に入るエンティティのベクトルのそれぞれの和を結合してパタンのベクトルを得ていた．これにより，片側のタイプが一致するだけで，パタンペアの類似度が高くなってしまったのだと考えられる．言い換えれば，パタンの意味を表すベクトルの構築には，エンティティを別々にではなく，エンティティペアの単位で用いる必要があると考えられる．なお，“X に ← 死去した → Y は” では “X に → 発表された → Y” のように，意味的にまったく無関係であるパタンを獲得してしまっているが，これも同様の理由によるものと考えられる．

APPROX-PMI+PCA が “X に ← 死去した → Y は” において誤って出力してしまったパタンは，「生まれる」と「死去」という，反意のパタンである．このように，反対の意味を持つものとの類似度が高くなってしまう事は，分布仮説を用いた手法では起こりやすく，また，解決し難い問題である．これを解決するために，人手で構築した単語の意味や，パタン内部の単語の意味なども利用し，パタンの意味を表現するベクトルを構築する必要があると考えられる．

4. 関連研究

教師なし関係抽出にはインスタンスやパタンの抽出，パタンの意味の表現，効率的な類似度計算という，三つの主要な問題がある．インスタンスやパタンの抽出法については，多くの手法が提案されてきた [1], [2], [5], [26] が，本研究ではこの点には取り組まないため，詳細についてここでは控える．

パタンの意味の表現方法として，PMI や LDA など様々な手法が利用されてきた [10], [14], [28]．Lin と Pantel [10] はエンティティとパタン間の共起情報として，PMI を用いてパタンの意味を表現した．彼らの目的は Relation Extraction ではなく言い換え表現（推論規則）の獲得であったが，この研究は教師なし関係抽出に大きな影響を与えた．Yao ら [28] はパタンとエンティティとの共起情報に加え，文や文書のテーマを LDA によってモデル化し，パタンの意味を表現するために利用した．また，近年，単語の意味などを表現するための学習法として，ニューラル言語モデルが注

[*8] 厳密には，Exact，Approx では連想配列上で次元とその値の両方を記憶する必要があり，次元の記憶のためにより多くのメモリが必要となる．

[*9] 評価用データ，すなわち，4,531 個のパタンペアのうち，類似度の高い順にペアを記載した．

手法	対象のパタン	上位 3 近傍のパタン
APPROX-COUNT +PCA	X を → 綴った → Y	X → 描いた → Y，X を → まとめた → Y X を → 書いた → Y
	X に ← 死去した → Y は	X に ← 生まれた → Y は*，X に → 生まれた → Y* X に ← 生まれる → Y は*
EXACT-PMI +WORD2VEC	X を → 綴った → Y	X を → 描いた → Y，X を → 扱った → Y X で ← 書いた → Y を*
	X に ← 死去した → Y は	X に → 発表された → Y*，X に → 発表した → Y* X には ← 発表 → Y を*

表 3 各手法での，対象についての上位 3 近傍のパタン

目を集めている [3], [13], [14], [15]．本研究では，共起頻度と PMI，ニューラル言語モデルによる単語の意味表現 [14] の比較を行う．

効率的な類似度計算を達成するために，エンティティのタイプ（例えば，“フランツ・カフカ” であれば「作家」というタイプに属する）を利用する研究が行われてきた [16], [17]．Min ら [16] は “X is Y”（X はエンティティ，Y はタイプ）という簡単なパタンや，エンティティの周辺単語などの情報を元にエンティティのクラスタリングを行い，得られたクラスタをタイプとして利用した．彼らはパタン間の類似度計算の際に，計算対象をエンティティのタイプが同一のパタンペアのみに制限し，計算時間を削減している．Nakashole ら [17] は Yago や Freebase などの既存の知識ベースから得たエンティティタイプの情報を元に，Min らと同様に，計算対象を制限し，計算時間削減を行った．しかしながら，タイプの粒度や，エンティティがどのタイプを表すかは文脈に依存するものであり，あらかじめ決定してしまう事は不適当であると考えられる．例えば “ウディ・アレン” は「映画監督」,「俳優」,「作家」であり，文脈によってどのタイプへの言及であるかが異なる．このため，タイプにより計算対象を削減するのではなく，パタンの意味表現を簡略化し，全ペアの計算も可能であるようにすることが望ましい．

本研究と最も近しいものは Goyal ら [7] による研究だろう．彼らは count-min sketch という近似頻度計算手法と近似最近傍探索手法を NLP のタスクに適用する効果を検証した．彼らは近似手法による素性ベクトルの獲得，ベクトルの次元の削減，クエリに対し類似のアイテム検索を行い，効果を示したが，個別の NLP タスクに関しては検証や有効性を評価していない．

5. まとめ

本研究では大規模データに対し，教師なし関係抽出を行うシステムを提案した．膨大な量のデータを処理するために，次元圧縮，近似頻度計算，単語のベクトルによる意味表現の三つの手法を利用した．実験を通して，近似頻度計算により，必要な情報を落とす事なく，省メモリでパタンの意味を表現するベクトルが作成できる事を示した．さらに，次元圧縮が類似度計算時間を短縮させるばかりではなく，パタンのベクトルの質を向上させもする事が明らかとなった．

これに対し，単語の意味表現ベクトルの利用では，良質のパタンのベクトルを得る事はできなかった．これは，エンティティペアをエンティティに分けてしまい，ペアとしての情報を利用できていなかったからと考えられる．これを解決するため，エンティティペアの分布を表現したようなパタンのベクトル表現を学習する必要があると推測される．今後の展望は，単語の意味表現ベクトルとパタンのベクトルを同時に学習するような手法を構築する事である．さらに，パタンに含まれる単語（パタンを構成する語）の情報もパタンの意味表現ベクトル内に組み込む手法も必要だろうと考えられる．

謝辞 本研究は，JST 戦略的創造研究推進事業「さきがけ」，および JST 戦略的創造研究推進事業「CREST」から部分的な支援を受けて行われた．

参考文献

[1] Akbik, A., Visengeriyeva, L., Herger, P., Hemsen, H. and Löser, A.: Unsupervised Discovery of Relations and Discriminative Extraction Patterns, *Proceedings of the 24th International Conference on Computational Linguistics*, pp. 17–32 (2012).

[2] Banko, M., Cafarella, M. J., Soderland, S., Broadhead, M. and Etzioni, O.: Open information extraction from the web, *IN IJCAI*, pp. 2670–2676 (2007).

[3] Bengio, Y., Ducharme, R., Vincent, P. and Janvin, C.: A Neural Probabilistic Language Model, Vol. 3, pp. 1137–1155 (2003).

[4] De Saeger, S., Torisawa, K., Kazama, J., Kuroda, K. and Murata, M.: Large Scale Relation Acquisition Using Class Dependent Patterns, *Proceedings of the 2009 Ninth IEEE International Conference on Data Mining*, pp. 764–769 (2009).

[5] Fader, A., Zettlemoyer, L. S. and Etzioni, O.: Paraphrase-Driven Learning for Open Question Answering., *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, The Association for Computer Linguistics, pp. 1608–1618 (2013).

[6] Firth, J. R.: *Papers in linguistics 1934-51*, Oxford University Press (1957).

[7] Goyal, A., Daumé, III, H. and Guerra, R.: Fast Large-scale

Approximate Graph Construction for NLP, *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 1069–1080 (2012).

[8] Halko, N., Martinsson, P. G. and Tropp, J. A.: Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions, *SIAM Review*, Vol. 53, No. 2, pp. 217–288 (2011).

[9] Hasegawa, T., Sekine, S. and Grishman, R.: Discovering Relations Among Named Entities from Large Corpora, *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, pp. 415–422 (2004).

[10] Lin, D. and Pantel, P.: DIRT - Discovery of Inference Rules from Text, *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 323–328 (2001).

[11] Mausam, Schmitz, M., Bart, R., Soderland, S. and Etzioni, O.: Open Language Learning for Information Extraction, *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 523–534 (2012).

[12] Metwally, A., Agrawal, D. and El Abbadi, A.: Efficient Computation of Frequent and Top-k Elements in Data Streams, *Proceedings of the 10th International Conference on Database Theory*, pp. 398–412 (2005).

[13] Mikolov, T., Karafiát, M., Burget, L., Cernocký, J. and Khudanpur, S.: Recurrent neural network based language model, *INTERSPEECH*, pp. 1045–1048 (2010).

[14] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S. and Dean, J.: Distributed Representations of Words and Phrases and their Compositionality, *Proceedings of the Neural Information Processing Systems Conference and Workshops*, pp. 3111–3119 (2013).

[15] Mikolov, T., V. Le, Q. and Sutskever, I.: Exploiting Similarities among Languages for Machine Translation, *CoRR*, Vol. abs/1309.4168 (2013).

[16] Min, B., Shi, S., Grishman, R. and Lin, C.-Y.: Ensemble Semantics for Large-scale Unsupervised Relation Extraction, *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 1027–1037 (2012).

[17] Nakashole, N., Weikum, G. and Suchanek, F.: PATTY: A Taxonomy of Relational Patterns with Semantic Types, *2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012)*, pp. 1135–1145 (2012).

[18] Pantel, P. and Lin, D.: Discovering Word Senses from Text, *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 613–619 (2002).

[19] Pantel, P. and Pennacchiotti, M.: Espresso: Leveraging Generic Patterns for Automatically Harvesting Semantic Relations, *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pp. 113–120 (2006).

[20] Ravichandran, D. and Hovy, E.: Learning Surface Text Patterns for a Question Answering System, *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pp. 41–47 (2002).

[21] Riedel, S., Yao, L., McCallum, A. and M. Marlin, B.: Relation Extraction with Matrix Factorization and Universal Schemas, *HLT-NAACL*, pp. 74–84 (2013).

[22] Riloff, E.: Automatically Generating Extraction Patterns from Untagged Text, *Proceedings of the Thirteenth National Conference on Artificial Intelligence - Volume 2*, pp. 1044–1049 (1996).

[23] Rosenfeld, B. and Feldman, R.: Clustering for Unsupervised Relation Identification, *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, pp. 411–418 (2007).

[24] Shinyama, Y. and Sekine, S.: Preemptive Information Extraction Using Unrestricted Relation Discovery, *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, Stroudsburg, PA, USA, Association for Computational Linguistics, pp. 304–311 (online), DOI: 10.3115/1220835.1220874 (2006).

[25] Szpektor, I., Tanev, H., Dagan, I. and Coppola, B.: Scaling Web-based Acquisition of Entailment Relations, *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pp. 41–48 (2004).

[26] Wu, F. and Weld, D. S.: Open Information Extraction Using Wikipedia, *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 118–127 (2010).

[27] Yao, L., Haghighi, A., Riedel, S. and McCallum, A.: Structured Relation Discovery using Generative Models, *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pp. 1456–1466 (2011).

[28] Yao, L., Riedel, S. and McCallum, A.: Unsupervised Relation Discovery with Sense Disambiguation, *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, pp. 712–720 (2012).