原著論文 Original Paper

Espresso 型ブートストラッピング法における意味ドリフトのグラフ理論に基づく分析

—語義曖昧性解消における評価—

Semantic Drift in Espresso-style Bootstrapping: Graph-theoretic Analysis and Evaluation in Word Sense Disambiguation

小町 守 Mamoru Komachi
奈良先端科学技術大学院大学 Nara Institute of Science and Technology
mamoru-k@is.naist.jp, http://cl.naist.jp/~mamoru-k/

工藤 拓 Taku Kudo
グーグル株式会社 Google Inc.
taku@google.com

新保 仁 Masashi Shimbo
奈良先端科学技術大学院大学 Nara Institute of Science and Technology
shimbo@is.naist.jp, http://cl.naist.jp/~shimbo/

松本 裕治 Yuji Matsumoto
（同 上）
matsu@is.naist.jp, http://cl.naist.jp/staff/matsu/home.html

keywords: Bootstrapping, Link Analysis, HITS, Regularized Laplacian, von Neumann Kernel, Word Sense Disambiguation, Semi-supervised Learning

Summary

Bootstrapping has a tendency, called *semantic drift*, to select instances unrelated to the seed instances as the iteration proceeds. We demonstrate the semantic drift of Espresso-style bootstrapping has the same root as the topic drift of Kleinberg's HITS, using a simplified graph-based reformulation of bootstrapping. We confirm that two graph-based algorithms, the von Neumann kernels and the regularized Laplacian, can reduce the effect of semantic drift in the task of word sense disambiguation (WSD) on Senseval-3 English Lexical Sample Task. Proposed algorithms achieve superior performance to Espresso and previous graph-based WSD methods, even though the proposed algorithms have less parameters and are easy to calibrate.

1. はじめに

近年自然言語処理では，機械学習を適用した手法が広く使われ成功を収めているが，これらの手法の多くは大規模なタグ付きコーパスを訓練データとして必要とする．タグ付きコーパスが整備されている言語は限られているため，人手によるタグ付けコストの削減は自然言語処理における重要な問題の一つである．

人手による資源作成のコストを最小限に抑えるため，半（弱）教師あり手法や教師なし手法が盛んに研究されている．本研究では特に自然言語処理における**ブートストラッピング**手法*1[Yarowsky 95, Abney 04] に注目し，分析を行う．本研究が対象とするブートストラッピング法は，獲得対象となるクラス（例:**is-a** 関係）のインスタンス（例: (cat, animal)）をシードとして与え，コーパスからインスタンスと共起するパターン（例: Y **such as** X）を抽出し，抽出した共起パターンを用いて新たなインスタンスを抽出する，といった手順を反復的に繰り返すことで，少数のインスタンスから大規模なインスタンス集合を再帰的に獲得する手法のことである．ブートストラッピング法は，語義曖昧性解消 [Yarowsky 95]，固有表現抽出 [Collins 99]，関係抽出 [Hearst 92, Riloff 99, Pantel 06] など自然言語処理のさまざまな分野で用いられている．

これらのブートストラッピング法では，反復処理を繰り返していくうちにシードインスタンスとは関係のないインスタンスを抽出してしまう問題が知られており，**意味ドリフト**[Curran 07] と呼ばれている．意味ドリフトは，ブートストラッピング法の反復過程において再現率（獲得すべきインスタンスのカバー率）は高いが適合率（共起するインスタンス中に目的のインスタンスが含まれている率）の低いパターン（**ジェネリックパターン**と呼ばれる）を抽出してしまうことに起因する現象である．たとえば，ウェブコーパスから観光地として有名な場所の名前を収集するタスクを考える．「ハワイ」といった単語

*1 統計学におけるブートストラップ（リサンプリングの一種）とは無関係なので注意．

をシードインスタンスとして与えると，ブートストラッピング法はやがて「○○の写真」や「○○の最新情報」といった，シードと関係のないインスタンスとも共起するパターンを抽出してしまう恐れがある．いったんこのようなジェネリックパターンを抽出してしまうと，以後の反復で「携帯電話」のように，シードとは無関係だが，出現頻度が高くジェネリックパターンともよく共起するインスタンスが抽出されてしまう．

意味ドリフトを回避するもっとも単純な方法は，ジェネリックパターンを抽出する前に反復を打ち切ってしまうことであるが，反復を停止する最適回数はタスクに依存し，事前に決定するのは困難である．近年考案された**Espresso アルゴリズム** [Pantel 06] は洗練されたスコアリング関数を用いて相互再帰的にインスタンスとパターンのスコアを定義し，ジェネリックパターンの影響を抑えている．とはいえ反復回数を適切に設定しなければ，Espresso においてもいまだ意味ドリフトの問題が存在する．

ブートストラッピング法のもう一つの欠点は，性能に影響するパラメータが多数存在する点である．たとえば反復の停止条件，各反復で選択するインスタンスやパターンの数，などのパラメータ設定により，大きく性能が変わる．そのため，これらのパラメータは各タスクに合わせて調整する必要がある．

本研究では，Espresso 型のブートストラッピング法を対象に，グラフ理論に基づいて分析を行う．過去ブートストラッピング法に意味ドリフトが起きることは知られていたが，それがなぜ起きるのか，どのようなインスタンスにドリフトするのか理論的に解析した研究はこれまでにない．

本論文は以下のように構成される．2 章で先行研究を概観したあと，3 章ではまず Simplified Espresso と呼ぶもっとも単純な Espresso 型のブートストラッピング法を導入する．Simplified Espresso は Espresso からいくつかのヒューリスティックを取り除いた簡略版で，数学的な取り扱いが可能である．この Simplified Espresso とリンク解析分野の全体重要度計算法の一つである HITS アルゴリズム [Kleinberg 99] との類似性から，ブートストラッピング法における意味ドリフトはリンク解析で見られるトピックドリフトと本質的に同じ現象であることが示唆される．また，HITS はシードインスタンスによらず同じスコアベクトルに収束することから，Simplified Espresso においても意味ドリフトは不可避であることも含意する．

オリジナルの Espresso も Simplified Espresso と同じ計算原理を共有しており，同じ問題を内包している．オリジナルの Espresso は意味ドリフトを回避するために Simplified Espresso にはないヒューリスティックを用いているが，これらのヒューリスティックは限定的な効果しかないことを 3·3 節で実験的に示す．

4 章では 3 章の議論を踏まえ、ブートストラッピング法の扱うタスクに対しては，リンク解析における全体重要度ではなく，個々の節点間の関連度を計算する処理が適切であることを指摘し，2 つのリンク解析的関連度算出法をこれらのタスクに適用することを提案する．これらグラフ理論に基づく手法はトピックドリフトが制御しやすい，または起きない性質を持つため，意味ドリフトにも頑健であると予想される．また，パラメータを一つしか持たず，扱いやすい利点もある．

5 章では Senseval-3 Lexical Sample Task の語義曖昧性解消タスクに提案手法を適用し，実際に意味ドリフトが抑えられることを示す．最後に 6 章で結論を述べる．

2. 関 連 研 究

2·1 ブートストラッピング法の概観

ブートストラッピング法は人手によるタグ付け作業を軽減するための一般的なフレームワークである．[Hearst 92] は 3 種類の人手で作成した語彙統語パターンを与え，上位下位関係（**is-a**）にある単語ペアを獲得するためのブートストラッピング手続きを考案した．

ブートストラッピング法はさまざまなタスクに適用されている．[Yarowsky 95] は教師あり手法に匹敵する性能の教師なし語義曖昧性解消システムを提案した．また，[Abney 04] はヒューリスティックに基づく Yarowsky のアルゴリズムを目的関数の最適化問題として定式化し，同じような目的関数の最適化問題として扱うことができる 7 種類のブートストラッピング法を提案した．

また，[Curran 07] は相互排他的ブートストラッピングと呼ばれる手法を提案した．この手法は複数の意味クラスの学習を目的としてインスタンス抽出を並列に実行し，それぞれのインスタンス抽出を相互に排他的に実行することにより，意味ドリフトを抑制する．また，ストップクラスと呼ばれる単語のリストを用意することでインスタンス抽出に制限をかける．ストップクラスとは，特定の意味クラスに属する単語で，意味ドリフトを引き起こすことが分かっている単語の集合のことである．しかしながら，ストップクラスはタスクごと，ドメインごとに異なり，効果的なストップクラスの単語リストを作成するためには人手によるチェックが不可欠である．

2·2 分析対象のブートストラッピング法

本研究で分析対象とするブートストラッピング法は，入力としてシードインスタンス集合を受け取り，**パターン抽出**と**インスタンス抽出**の 2 フェーズを繰り返す．こうしたアルゴリズムとしては，[Yarowsky 95, Abney 04, Pantel 06, Curran 07] がある．たとえば *is-a* 関係 (X is-a Y) にある単語ペア (X, Y) を獲得するタスクを考える．シードとして (cat, animal) を与えると，パターン抽出フェーズでは (cat, animal) にマッチする可能なパターンの中から，Y *such as* X のように学習対象のインスタンスとよく共起する（一般に少数の）パターンを選択する．インスタ

ンス抽出フェーズでは，選択されたパターンと共起する新しいインスタンス，例えば (sparrow, bird) を獲得する．各フェーズの処理内容について述べる．

フェーズ 1. パターン抽出：シードインスタンスを手がかりに，コーパスから共起パターンを列挙する．典型的なパターンとしては，表層の文字列や係り受け関係のような語彙統語パターンが用いられる．各々のパターンに対して，シードとの共起頻度などに基づいて信頼度スコアを割り当て，スコア上位のパターンのみを選抜する（この処理を指して以下，**パターン抽出**と呼ぶ）．シードインスタンスと関連性が高いパターンにより大きなスコアを割り当て，ジェネリックパターンに大きなスコアを割り当てないようにスコア関数を設計する必要がある．

フェーズ 2. インスタンス抽出：フェーズ 1 で抽出したパターンと共起するインスタンスをコーパスから抜き出して列挙する．列挙したインスタンスの信頼度スコアを計算し，スコア上位のインスタンスを次の反復でシードインスタンスとして利用するために記憶する．この処理を**インスタンス抽出**と呼ぶ．ここで抽出したインスタンスを，(a) 各反復ごとに**獲得インスタンス**として出力するのか，あるいは，(b) 単に次の反復のシードとしてのみ用い，システムの出力は最終反復の際の信頼度スコアが上位のインスタンスとするのか，は，手法によって異なる．Espresso は後者 (b) である．

ブートストラッピング法は停止条件が満たされるまで上記の 2 フェーズを繰り返すが，反復が進むにつれて抽出インスタンスは誤りを含みやすくなるため，意味ドリフトが起きる前に停止することが重要である．

ブートストラッピング法で設定すべきパラメータとして，フェーズ 1 においては (i) **パターンの信頼度スコア関数**，(ii) **インスタンス抽出に使うパターンの個数**，フェーズ 2 においては，(iii) **インスタンスの信頼度スコア関数**，(iv) **各反復で抽出する（次の反復でシードとして用いる）インスタンスの個数**，がある．いつ反復を打ち切るか，を判定するための (v) **停止条件**もパラメータの一つである．

ブートストラッピング法の大きな弱点として，これら (i)–(v) の設定について，最適な設定やパラメータ値を選択する標準的な方法がないことがある [Ng 03]．また，意味ドリフトの問題もある．意味ドリフトの問題にどのように対処するかは個々のブートストラッピング法で異なっており，特定のタスクに特化したヒューリスティックが用いられることが多い．

2·3 Espresso アルゴリズム

最新のブートストラッピング法の一つである Espresso アルゴリズム [Pantel 06] が意味ドリフトにどのように対処しているか見てみる．従来法では意味ドリフトを抑えるためにジェネリックパターンを排除し，たかだか数十の適合率の高いパターンのみでインスタンス抽出を行っていたが，ジェネリックパターンの排除のためにパターンのフィルタリングを強く行っていることから，全体で獲得できるインスタンスが少なく，結果として再現率が十分でないという欠点があった．これに対して Espresso では巧妙な信頼度スコア関数を導入してジェネリックパターンの影響力を減らし，小さな重みをつけた全パターンのスコアの線形和を用いてインスタンスのスコアを計算する．この処理で再現率が向上し，全体で獲得できるインスタンス数が大幅に増加する．

Espresso の信頼度スコア計算の基本的なアイデアは，相互再帰的にインスタンスのスコアとパターンのスコアを定義する，ということである．つまり，信頼度の高いパターンとよく共起するインスタンスは信頼度が高く，逆に信頼度の高いインスタンスとよく共起するパターンも信頼度が高い，という考え方である．パターン p とインスタンス i のスコアはそれぞれ $r_\pi(p)$ と $r_\iota(i)$ であり，以下のように与えられる:

$$r_\pi(p) = \frac{1}{|I|}\sum_{i\in I}\frac{pmi(i,p)}{\max\ pmi} r_\iota(i) \tag{1}$$

$$r_\iota(i) = \frac{1}{|P|}\sum_{p\in P}\frac{pmi(i,p)}{\max\ pmi} r_\pi(p) \tag{2}$$

ここで

$$pmi(i,p) = \log_2 \frac{|i,p|}{|i,*||*,p|} \tag{3}$$

とする*2．P と I はそれぞれパターンとインスタンスの集合であり，$|P|$ と $|I|$ はパターンとインスタンスの数である．$|i,*|$ はコーパス内におけるインスタンス i の，$|*,p|$ は同じくパターン p の頻度である．$|i,p|$ はパターン p とインスタンス i が共起する回数である．また，$\max\ pmi$ は全てのインスタンスとパターンの組み合わせの間での *pmi* の最大値である*3．

Espresso は式 (1) をパターンのスコアリング関数（2·2 節のパラメータ (i)）に，式 (2) をインスタンスのスコアリング関数（同パラメータ (iii)）にそれぞれ用いる．その他のパラメータはタスクごとに違うので，調整が必要である．Espresso は式 (1)–(2) の信頼度スコアを用いることにより，適合率を高く保ちつつ再現率を大幅に向上させることに成功した．

3. Espresso 型ブートストラッピング法の分析

3·1 Simplified Espresso

意味ドリフトの原因を明らかにするために，図 1 に示す単純化したブートストラッピング法を考える．

*2 自己相互情報量は $\log_2 \frac{P(i,p)}{P(i)P(p)}$ であるが，彼らは Google の検索ヒット回数を用いて *pmi* を定義したため，コーパスのサイズを計算することができず，このように近似している．我々の実験の場合，コーパスのサイズは既知のため，*pmi* として自己相互情報量を用いた．

*3 この値は信頼度が発散しないよう正規化するために使われる．

入力: シードベクトル i_0
入力: パターン・インスタンス共起行列 M
出力: インスタンススコアベクトル i
出力: パターンスコアベクトル p

1: $i = i_0$
2: **repeat**
3: $p \leftarrow Mi$
4: p を正規化する
5: $i \leftarrow M^T p$
6: i を正規化する
7: **until** i と p が両方収束
8: **return** i と p

図 1 単純化したブートストラッピング法

Espresso の場合と同様，$|I|$ と $|P|$ をそれぞれインスタンスとパターンの数とする．提案するアルゴリズムはシードベクトル i_0 とパターン・インスタンス共起行列 M を入力とする．i_0 はシードインスタンスに対応する位置に 1，それ以外に 0 が入った $|I|$ 次元のベクトルである．M は $|P| \times |I|$ 次元の行列で，(p,i)-要素 $[M]_{pi}$ はパターン p とインスタンス i のコーパス内における（重みつき）共起スコアである．アルゴリズムではステップ 2–8 のループを繰り返し，i と p のいずれも収束したとき，i と p のペアを出力し終了する．

このアルゴリズムは単純ながら，Espresso の更新式 (1) と (2) をステップ 3–6 で表現することができる．つまり

$$[M]_{pi} = \frac{pmi(i,p)}{\max pmi}, \tag{4}$$

とし，ステップ 4 と 6 で

$$p \leftarrow p/|I| \quad \text{and} \quad i \leftarrow i/|P| \tag{5}$$

により p と i を正規化すれば，このアルゴリズムは Espresso のスコアの更新式と一致する．

上のように M として式 (4) を用い，ステップ 4，6 の正規化を式 (5) によって特殊化した図 1 のアルゴリズムを以後 **Simplified Espresso** と呼ぶ．Simplified（簡略版）の名前が示す通り，これは反復を収束するまで続け，全インスタンスを次の反復でのパターン計算に用いた Espresso の一種である．

3·2 リンク解析としての Simplified Espresso

n をステップ 2–10 の反復回数とする．式 (4) と (5) をステップ 3–6 に代入すると，n 回の反復後のインスタンスのスコアベクトルは

$$i_n = A^n i_0 \tag{6}$$

となる．ただし，

$$A = \frac{1}{|I||P|} M^T M \tag{7}$$

である．

行列 A が既約である，つまり A を隣接行列とするグラフが連結であるとする．i_n を毎回の反復で正規化しつつ n を増加させると，i_n は A の主固有ベクトルに漸近する．言い換えると，どのようなシードインスタンスを入力としても，収束するまで反復を繰り返すと，このアルゴリズムは同一のインスタンスランキングを返す．また，$A = \frac{1}{|I||P|} M^T M$ なので，A の主固有ベクトルは M を隣接行列と見なした（インスタンス，パターンの）二部グラフに対して，HITS アルゴリズム [Kleinberg 99] が返す権威度ベクトルに等しい*4．特筆すべき点として，式 (1)，(2) と HITS の類似性については [Pantel 06] では議論されていない．

A の主固有ベクトルはシードベクトル i_0 に依存しない．従って，上の議論は Simplified Espresso において意味ドリフトが不可避であることを示している．HITS においてもクエリによらず支配的なトピックに関連するページが高くランクされるという現象が知られており，**トピックドリフト**と呼ばれている [Bharat 98]．

しかしながら，HITS や Simplified Espresso とは異なり，Espresso や他のブートストラッピング法 [Yarowsky 95, Riloff 99] では信頼度の高いパターンとインスタンスのみをフィルタして次の反復で用いる．このヒューリスティック（以下、**フィルタリング**と呼ぶ）の効果について，次に実験で検証する．

3·3 Espresso の収束過程の解析

オリジナルの Espresso にはあって，Simplified Espresso では取り除かれた「各反復でもっとも信頼度の高いパターン・インスタンスのみを抽出する」フィルタリング処理が，意味ドリフトにどのような影響を与えているのか調べるため，それぞれのアルゴリズムを語義曖昧性解消タスクに適用した．

語義曖昧性解消は，正解語義が分かっている訓練データを用いて，与えられたテストインスタンスの正解語義（システムからは隠されている）を予測するタスクである．

テストインスタンスを i とすると，この実験で i の語義は以下の手順で予測される．

(1) 全インスタンス・全パターンを用い，パターン・インスタンス共起行列 M を構築する．用いたパターンの詳細については 5 章冒頭で述べる．
(2) テストインスタンス i のみをシードインスタンスとして，Simplified Espresso と Espresso を実行する．
(3) アルゴリズムが停止したあと，信頼度スコアが最上位の k 個の訓練インスタンスを選択する*5．

*4 i_n の各要素の相対的な大きさが変わらないかぎり，このベクトルは各反復の際どのように正規化してもよい．従って，HITS と Simplified Espresso は異なる正規化を用いているが，両方とも A の主固有ベクトルに収束する．
*5 本実験では $k = 3$ とした．

(4) 選択された k 個の訓練インスタンスの正解語義から，多数決でインスタンス i の予測語義を決定し，出力する．もし同数であった場合，i の中でもっともスコアが高いインスタンスの語義を出力する．

このように，ステップ (3) までは教師なしで実行でき，ステップ (4) だけが語義情報にアクセスする．ステップ (3), (4) は Espresso, Simplified Espresso 停止時のインスタンス信頼度スコアを近傍尺度とした k 最近傍法 [Cover 76] と見なせる[*6]．k 最近傍法は，単純な手法ではあるが，語義曖昧性解消タスクで高い性能 [Ng 97] を示すと報告されている．

ステップ (2) からわかるように，語義曖昧性解消タスクでは，曖昧性を解消する対象のインスタンスをシードとして用いる．従って，関係抽出タスクや固有表現抽出タスクと異なり，シードとして何を選択するか，に関する任意性がなく，シードインスタンスの影響をアルゴリズムの性能評価から排除することができ，都合が良い．

実験には，Senseval-3 Lexical Sample (S3LS) タスクのデータセット[*7]と，訓練データ・テストデータ分割をそのまま用いた．

本節では個々のアルゴリズムの典型的な振る舞いを観察するため，特定の一単語 *bank* の語義曖昧性解消を行った．全ての単語に対する性能は 5 章で調査する．S3LS データセットには 394 インスタンスの単語 *bank* が含まれており，それぞれの出現文脈と，正解の語義が与えられている．*bank* の 10 個の語義のうち，最頻出語義は「銀行」の意味の語義である．

この実験において，Espresso はインスタンススコアベクトルにおけるスコア上位 100 インスタンスのみをパターンスコア計算に用い，残りのスコアは 0 にクリアする．また，ゼロクリアしないで残すスコア上位のインスタンス数は，毎回の反復で 100 ずつ増加させる．一方，パターンスコアベクトルについては，パターンスコアベクトルの上位 200 パターン[*8]のみをインスタンススコア計算に用い，残りのスコアは 0 にクリアする．これら 3 つのパラメータ設定は予備実験を行い調整したため，Espresso 元論文 [Pantel 06] で用いられていた値とは異なる．一方，[Pantel 06] と同様，クリアしないスコア上位のパターン数は毎回の反復で 1 ずつ増加させる．

その他のパラメータは Simplified Espresso とオリジナルの Espresso とで共通とした．具体的にはスコア関数として式 (1) と (2) を用い，インスタンスは最終的なスコアによって獲得する．

*6 本実験の目的は適合率の高い語義曖昧性解消器を作ることではないため単純な手法を用いたが，適合率を上げるためには，ブートストラップ適用後の各インスタンスのスコアの重み付き多数決や，インスタンスのスコアを素性の一つとした Support Vector Machines による分類など，他の手法も考えられる．

*7 http://www.senseval.org/senseval3/data.html

*8 語義曖昧性解消タスクでは関係抽出や固有表現抽出と比較してデータスパースネスの問題が深刻なため，初期パターン数は比較的大きく設定されている．

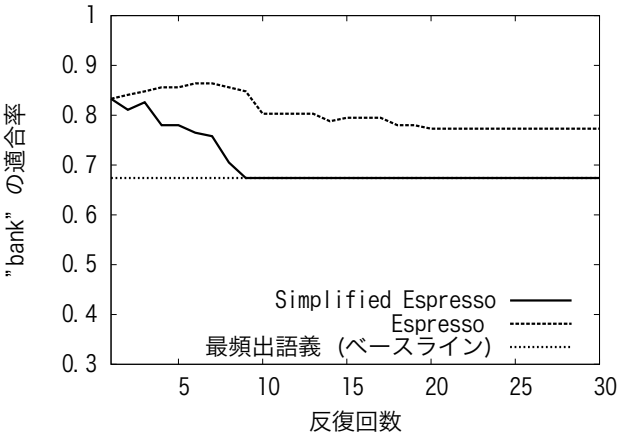

図 2 Simplified Espresso と Espresso の適合率

なお，[Pantel 06] は Espresso で関係抽出タスクを行う際の停止条件を示しているが，この条件は語義曖昧性解消をはじめ他のタスクにそのまま適用可能な一般性がない．このため以下の実験では停止条件を明示的に定めず，収束するまで反復を繰り返し，各反復後の適合率を観察した．

図 2 は Simplified Espresso と Espresso の収束過程を示している．X 軸はブートストラッピングの反復回数を，Y 軸は適合率（= システム出力中の正解事例数 / システムの全出力事例数）を表す．

Simplified Espresso は反復回数が増えるに従って最頻出語義（「銀行」の意味）を多く出力するようになり，9 回の反復後はシードインスタンスによらず，最頻出語義を選択するようになった．3·2 節における議論から予想されたように，反復が進むにつれて意味ドリフトが起きている．実際，収束したあとのインスタンスのランキングを検証してみたところ，パターン・インスタンス行列 M から計算した HITS 権威度ランキング（M^TM の主固有ベクトルによって導かれるランキング）と一致した．

一方，Espresso は意味ドリフトの影響が比較的小さい．最終的に 20 回の反復を繰り返して収束したあとの適合率は 0.773 であり，最頻出語義を用いるベースラインを 0.10 ポイント上回っている．しかしながら，詳細に検討してみると，Espresso のフィルタリングのヒューリスティックの効果は限定的であることが分かる．

図 3 は正解語義が最頻出語義「銀行」であるものと，それ以外の語義であるものに分け，各々をどれだけ当てられたか（= 再現率），を反復回数ごとに描いたものである．意味ドリフトが起きていれば，反復回数が増えるに従い最頻出語義として予測されるインスタンスの数が増え，最頻出語義の再現率が上昇する一方，その他の語義の再現率が低下するはずである．図 3 はまさにその傾向を捉えており，Espresso も意味ドリフトの影響から完全に逃れることができていない．またその結果，7 回目の反復以降 Espresso の適合率が下降していることが図 2

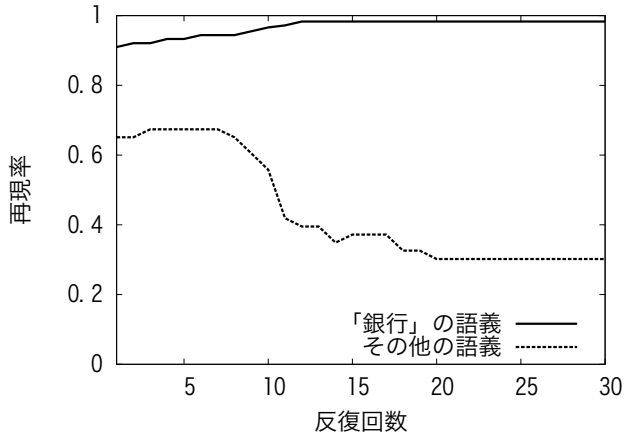

図 3 最頻出語義「銀行」と，それ以外の語義を正解とするインスタンスごとの Espresso の再現率

からも見て取れる．

4. 意味ドリフトを解決するための 2 つのグラフ理論に基づくアルゴリズム

3 章で議論したとおり，Espresso 型のブートストラッピング法における意味ドリフトは，その計算方法がリンク解析における「全体重要度（具体的には HITS 権威度)」算出法に類似していることが原因と考えられる．われわれは，ブートストラッピング法が扱うタスクは「重要度算出」ではなく，シードとして与えられた節点と，他の節点との「関連度（あるいは類似度)」を算出するタスクであることに着目し，このタスクに対して，リンク解析分野で提案された 2 つの関連度算出法を適用することを提案する．これらの手法は，トピックドリフトが制御しやすい，あるいは起きない性質を持つことが知られているため，意味ドリフトに対しても頑健である．また，ここで提案する 2 つの手法はブートストラッピング法と比較してパラメータ数が少なく，扱いが容易である．

4·1 von Neumann カーネル

[Kandola 02] は単語を用いて文書の類似度を測るため，**von Neumann カーネル**を提案した．[Ito 05] は von Neumann カーネルの性質について分析し，von Neumann カーネルが共引用解析における関連度と HITS の権威度との混合（相対的重要度）と見なせ，そのパラメータ（拡散係数）によってそれらのトレードオフを制御できることを示した．

Espresso の成功の要因は，HITS に類似した相互再帰的なインスタンスとパターンの信頼度の定義と，同じく HITS と同様な繰り返し計算によって高次の相関を考慮し再現率を確保した点にある．しかしながらこの HITS との類似性は，前節で見た通り意味ドリフトの原因ともなっている．また，ブートストラッピング法が対象とするタスクは，HITS のような全体重要度算出よりも個々のシードに対する関連度計算に基づくべきだ，という直感がある．そこで，本節では von Neumann カーネルが HITS と共引用関連度の中間（相対的重要度）と見なせる点に着目し，これによって意味ドリフトの抑制を試みる．von Neumann カーネルを文書・単語行列ではなくパターン・インスタンス共起行列から計算すれば，任意のインスタンスのシードインスタンスに対する相対的重要度を推定することができる．

具体的には，インスタンス類似度行列 A をパターン・インスタンス共起行列 M を用いて $A = M^T M$ とし，λ を A の主固有値とする．拡散係数を $\beta\,(0 \leq \beta < \lambda^{-1})$ とする **von Neumann カーネル行列** K_β は以下のように定義される:

$$K_\beta = A\sum_{n=0}^{\infty}\beta^n A^n = A(I-\beta A)^{-1}. \qquad (8)$$

2 つのインスタンス i,j 間の類似度は K_β の (i,j) 要素で与えられる．ここで，i-列目のベクトルはシードインスタンス i に対するスコアベクトルとして用いることができる．また，複数のシードを用いるときは該当する各行の和を取ったスコアベクトルを用いる．

式 (8) 中の各項 $A^n\,(=(M^TM)^n)$ の (i,j) 要素は，$A(=M^TM)$ を隣接行列とするインスタンス共起グラフにおける，節点（インスタンス）i から j に至る長さ n の経路の総数を表す．$n=1$ に対応する (M^TM) はシードインスタンスに対する共引用解析関連度を示す．一方，n が十分大きいとき，$(M^TM)^n$ の各行（あるいは列）の与えるインスタンスのランキングは Kleinberg の HITS 権威度に近づく．従って，拡散係数 β を小さく設定すれば関連度に従ったランキングとなり，拡散係数 β を大きく設定すれば HITS 権威度に従ったランキングとなる．

結論として，拡散係数 β を適切な値に設定できれば，最頻出語義とその他の語義とのバランスの取れた類似度計算ができると予想される．5·3 節 で，この予想が正しいことを検証する．

4·2 正則化ラプラシアンカーネル

von Neumann カーネルは関連度と重要度の混合と見なすことができ，パラメータ（拡散係数）β によって関連度と重要度のトレードオフが制御可能である．パラメータが一つだけなので，その調整作業はブートストラッピング法と比べて格段に容易とはいえ，やはり使用者にとって負担である．リンク解析分野で提案されたもう一つの関連度計算法である，正則化ラプラシアンカーネル [Smola 03, Chebotarev 98] は，von Neumann カーネル同様の拡散係数パラメータを持つが，このパラメータに対して安定しており，パラメータ調整はより簡単と考えられる．また，von Neumann カーネルと異なり，拡散係数を大きくしても重要度尺度とならず，関連度としての性質を保つ．

この点もブートストラッピング法が対象とするタスクの性質を考えると好ましい．実際，このカーネルは $\beta \to \infty$ において均一行列（全ての節点が同じように関連していることを示す行列）に収束する [Ito 05].

G を重み付き無向グラフとし，その隣接（対称）行列を A とする．グラフ G の（正規化）ラプラシアン行列 $\mathcal{L}$ は，

$$\mathcal{L} = I - D^{-1/2} A D^{-1/2} \tag{9}$$

と定義される．ここで, D は i 番目の対角要素 $[D]_{ii}$ として節点 i の次数を持つ対角行列である．すなわち，$[A]_{ij}$ を A の (i,j) 要素として $[D]_{ii} = \sum_j [A]_{ij}$．式 (8) の A を $-\mathcal{L}$ で置き換え，最初の A を削除すると，**正則化ラプラシアンカーネル**を得る．

$$R_\beta = \sum_{n=0}^{\infty} \beta^n (-\mathcal{L})^n = (I + \beta\mathcal{L})^{-1} \tag{10}$$

ここでも $\beta(\geq 0)$ は拡散係数と呼ばれるパラメータである．

正則化ラプラシアンも von Neumann カーネル同様，高次の項（ただし，正則化ラプラシアンの場合は A^n $(= (M^T M)^n)$ ではなく $(-\mathcal{L})^n = (D^{-1/2} A D^{-1/2} - I)^n)$ を内包しており，インスタンス間の高次の相関を考慮した尺度となっていることがわかる．

von Neumann カーネルとの大きな違いは，A の代わりに $D^{-1/2} A D^{-1/2}$ によって各節点に向かう辺の重みを節点の次数で正規化している点にある．(ジェネリックパターンと共起し）次数の高い節点（インスタンス）に接する辺の重みがより大きく減じられることから，ジェネリックパターンの影響は強く抑制される．

5. 語義曖昧性解消実験

von Neumann カーネル，正則化ラプラシアンカーネルを S3LS の語義曖昧性解消タスクで評価した．カーネル計算に用いるパターン・インスタンス共起行列行列 M には Simplified Espresso と同じ式 (4) を用いた．パターンとしては，以下の 2 種類を用いた．

bag-of-words パターン: S3LS データセットで提供されている文脈（パラグラフ）に出現する全単語を bag-of-words パターン（1 グラム）として用いた．各単語が 1 パターンを構成する．インスタンス i の文脈で単語 w が出現した場合，そのパターンが 1 に設定される．単語は全て小文字に変換し，Porter Stemmer[*9]を用いた前処理を行った．

局所的な共起パターン: 局所的な共起パターンとは，インスタンス周辺の単語列からなる局所的な文脈を指す．たとえばインスタンス *interest* に対し「sale of * *interest* in * *」などがパターンとして用いられる（「*」はワイルドカードであり，任意の単語が入る．たとえば上記のパターンは「sale of their *interest* in Mandarin Oriental」にマッチする）．予備実験により，ウィンドウサイズは ±3 に設定した．

拡散係数 β は von Neumann カーネルにおいては 10^{-5}, 正則化ラプラシアンカーネルにおいては 10^{-2} に設定した[*10].

表 1 *bank* に対する正解語義別の再現率

アルゴリズム	最頻出語義	その他
Simplified Espresso	100.0	0.0
Espresso（収束後）	100.0	30.2
Espresso（最適反復回数）	94.4	67.4
von Neumann カーネル ($\beta = 10^{-5}$)	92.1	65.1
正則化ラプラシアン ($\beta = 10^{-2}$)	92.1	62.8

5·1 実験 1: 意味ドリフトの抑制

まず von Neumann カーネルと正則化ラプラシアンカーネルにおいて実際に意味ドリフトが抑制できているかを調べるため，これらカーネルを 3·3 節 と同じ単語 *bank* の語義曖昧性解消タスクに適用した．実験設定は 3·3 節と同じである．3·3 節の図 3 同様，最頻出語義が正解であるインスタンスに対する再現率と，それ以外のインスタンスに対する再現率の二つで評価した．意味ドリフトが起きているときにシステムはもっぱら最頻出語義を出力するようになり，その他の語義を正解とするインスタンスの再現率が低下することが予測される．

表 1 は結果である．比較対象に用いた Espresso と Simplified Espresso については，3·3 節の図 3 のグラフに示した結果を数値化して掲載してある．表中の「Espresso（最適反復回数で停止)」は，*bank* インスタンス全体の適合率が最高となった反復回数（7 回; 図 2 参照）で Espresso を打ち切った際の性能を示している．

二つのカーネルはいずれも最頻出語義とその他の語義の双方において，Espresso を最適な反復回数で打ち切った場合とほぼ同等の再現率を達成している．Simplified Espresso や，Espresso を収束するまで実行した場合とは異なり，最頻出語義以外を正解とするインスタンスに対する再現率の低下が見られず，意味ドリフトが抑制されていると考えられる．

5·2 実験 2: 語義曖昧性解消評価タスク

提案した二つのカーネルの有効性を示すため，単語 *bank* だけではなく，S3LS データセットの中にある全名詞を対象にした語義曖昧性解消実験と，全データ（名詞・動詞・形容詞）を対象にした実験を行った．

*9 http://tartarus.org/˜martin/PorterStemmer/def.txt

*10 4 章で述べたように，von Neumann カーネルにおいては拡散係数を小さく設定したほうが適合率が高く，正則化ラプラシアンカーネルにおいては拡散係数による影響をあまり受けないと考えられる．5·3 節 で後述するが，これらの値はそれぞれ von Neumann カーネルと正則化ラプラシアンカーネルいずれにおいても適合率が最大となる拡散係数を選択した．

表 2 語義曖昧性解消アルゴリズムの適合率

アルゴリズム	名詞	全体
最頻出語義	54.5	55.2
HyperLex [Véronis 04]	64.6	–
PageRank [Agirre 06]	64.5	–
Simplified Espresso	44.1	42.8
Espresso（収束後）	46.9	59.1
Espresso（最適反復回数で停止）	66.5	63.6
von Neumann カーネル ($\beta = 10^{-5}$)	**67.2**	64.9
正則化ラプラシアンカーネル ($\beta = 10^{-2}$)	67.1	**65.4**

表 2 に実験結果を示す．ベースラインとして，インスタンスによらず常に最頻出語義を出力した場合の適合率を用い，比較対象として，Espresso (収束後, および最適な反復回数で停止した場合）と Simplified Espresso を用いた．この実験で Espresso は 20 回の反復後収束し，7 回目に適合率が最高となった（表中の「Espresso（最適反復回数で停止）」）．また，参考として同じデータに対して [Agirre 06] で報告されている二つの（グラフ理論的）手法の結果（名詞のみ）を掲載した．これら手法は，曖昧性解消対象の単語と共起するパターンの共起グラフを作成し，そのグラフにおけるハブを抽出したあと，最小全域木に基づくアルゴリズムを適用して語義曖昧性解消を行う．ハブの抽出の際に HyperLex[Véronis 04] と PageRank[Brin 98] というグラフに基づく 2 つのアルゴリズムが用いられており，いずれを用いた場合も，同じ素性を用いた教師あり手法と同程度の適合率が得られると報告されている．

表 2 を見ると，von Neumann カーネル，正則化ラプラシアンの適合率は，他のいずれの手法をも上回っていることがわかる．提案手法と Espresso（最適反復回数で停止）の適合率の差は，名詞において 1 ポイント未満であり，ほぼ同等ではあるが，Espresso のこの数値は，最適な反復回数で試行を打ち切った場合の性能であることに注意．一般に，そのような停止条件を正確に求めることはきわめて難しい．提案手法は，Agirre らの HyperLex と PageRank を用いた手法に対してもそれぞれ 2.5 ポイント以上上回る適合率を示している．彼らの手法は最適な性能を得るために 7 つのパラメータを調整しなければならないのに対し，提案手法の持つパラメータは一つだけであり，特に正則化ラプラシアンはパラメータに対して安定しているため調整が容易である．

なお，最適反復回数で停止しない Espresso は最適反復回数で停止したときと比べて, 名詞を対象にした場合約 20 ポイント, 全品詞を対象にした場合 4 ポイント低い適合率であった．特に名詞のみを対象にした場合の性能は Simplified Espresso と大差ない．このことから，Espresso においては，フィルタリングに加え最適反復回数での停止もまた意味ドリフトを抑制するために必須の処理であると推測される．この結果は 3·3 節における Espresso の収束プロセスの解析実験の結果とも合致している．

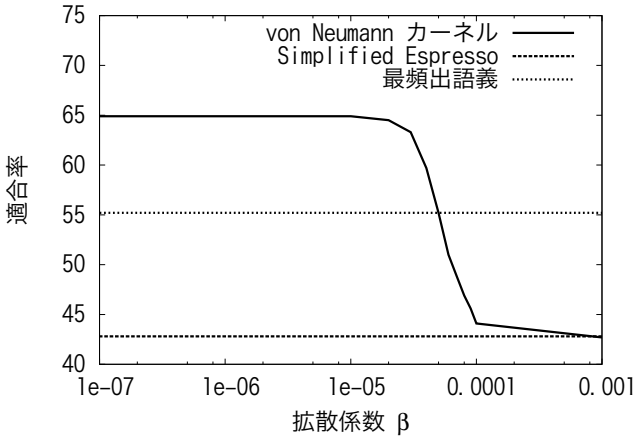

図 4 S3LS WSD タスクにおける異なる拡散係数 β に対する von Neumann カーネルの適合率

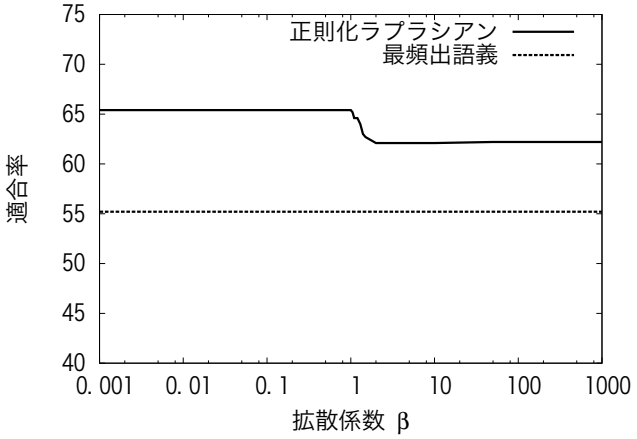

図 5 S3LS WSD タスクにおける異なる拡散係数 β に対する正則化ラプラシアンの適合率

また，3·3 節の実験では，Simplified Espresso は収束後最頻出語義を出力した．しかしながら，本実験においては Simplified Espresso は最頻出語義のベースラインより 10 ポイント適合率が低い．Simplified Espresso の収束先は A の主固有ベクトルであり，このベクトルが常に最頻出語義を反映したものになっているとは（多くの場合これらは一致することが想定されるが）限らないため，このような不一致が起きている[*11]．

5·3 実験 3: 拡散係数に対する安定性

拡散係数 β に対する二つのカーネルの依存性を見るために，β の値を変えた場合の S3LS の全データに対する

*11 似ているがより極端な例として [Ito 05] では正規化された共起行列 M を用いると直観に反する主固有ベクトルに収束することが示されている．本実験においては M の各要素に自己相互情報量を用いたので，直接の頻度を各要素の値として用いる場合の収束先とは異なった結果になったと考えられる．

適合率を図 4 と図 5 に示した．von Neumann カーネル（図 4）は β が大きくなるにつれ漸近的に HITS の権威度ランキング（=Simplified Espresso）に近づくため，予想されたように意味ドリフトの影響を受け適合率が低下している．また，この低下は $10^{-5} < \beta < 10^{-4}$ にかけて急激に起きている．これに対し図 5 の正則化ラプラシアンカーネルは β の値に対してほぼ安定している．

5·4 議　　論

語義曖昧性解消タスクや固有表現抽出タスクといったタスクでは，パターンの曖昧性が高く，なおかつ抽出に用いられるパターンの数が多いので，全体重要度に基づく手法 (HITS に類似した Simplified Espresso) では意味ドリフトが起きやすいと考えられる．ブートストラッピング法は数十回反復を繰り返すと収束してしまうため，各タスクで最適なパターン数の初期値や毎回の反復でスコアをゼロにクリアしないパターンの数といったパラメータを調整するのは面倒である．そのため，こういったタスクに関してはグラフに基づく関連度算出手法が効果的と予想される．実際，固有表現抽出タスクに類似した検索クエリ分類タスク [Li 08] では self-training（ブートストラッピング法）よりグラフを用いた手法のほうがコンスタントに性能が高いと報告されている．

一方，関係抽出のような 2 項関係をインスタンスとして抽出するタスクでは，パターンの曖昧性が低く，抽出に用いられるパターンの数が少ないため，グラフが疎になり，たとえば A が連結でない場合，全体重要度に基づく手法でも意味ドリフトが起きない可能性がある．本論文では，意味ドリフトが起きやすいと考えられる語義曖昧性解消によって，2 つのグラフカーネルに基づく関連度算出手法がドリフトを防げることを検証したが，グラフの構造によっては全体重要度に基づく手法でも十分な場合が考えられる．

また，5·3 節で見たように，von Neumann カーネルの正則化ラプラシアンに対する欠点として，拡散係数 β に対する過敏性があった．とはいえ，von Neumann カーネルにおける拡散係数 β は A の主固有値の逆数 $1/\lambda$ によって上限が定まり，β によるカーネル行列の微分を用いることで効率的に β を調整することができる（詳細は [Ito 05] を参照）．最近 [Kunegis 09] は von Neumann カーネル，正則化ラプラシアンを含むさまざまなグラフカーネルに対するパラメータ調整を自動的に行う方法を提案している．こういった自動化が実現できるのは，これらカーネルがヒューリスティックによらないグラフ理論に基づく手法であり，理論的な分析が可能であったことによるところが大きい．

6. ま　と　め

本論文では Espresso 型のブートストラッピング法における意味ドリフトに対し，グラフ理論に基づく分析を与えた．ブートストラッピング法における意味ドリフトは HITS におけるトピックドリフトに対応する概念であることを指摘した．リンク解析で用いられている von Neumann カーネルと正則化ラプラシアンが意味ドリフトを抑制することを Senseval-3 Lexical Sample タスクによって示した．提案手法は 1 つのパラメータしか持たず，調整が容易である．

ブートストラッピング法が用いられる典型的な自然言語処理タスクは語義曖昧性解消と関係抽出，そして固有表現抽出である．意味ドリフトの起きやすい固有表現抽出タスクに関して，[Curran 07] らの相互排他的ブートストラッピング法は意味ドリフトを起こすようなインスタンス（ストップクラス）を人手で指定し，適合率の高いブートストラッピング法を行うアルゴリズムだが，意味ドリフトの原因がトピックドリフトであることが明らかになったため，パターン・インスタンス共起行列から自動でストップクラスに属するインスタンスを獲得することが可能である．シードインスタンスの選択の半自動化 [Vyas 09] とともに，ブートストラッピング法における人手の編集作業の削減が考えられる．

リンク解析分野では，正則化ラプラシアン以外にも，ラプラシアン行列における固有値の正則化方法に応じてさまざまなグラフカーネルが提案されている [Kondor 02, Nadler 06, Saerens 04]．特に，ラプラシアンの疑似逆行列を用いて定義される通勤時間カーネル [Saerens 04] は一つもパラメータを持たないが，協調フィルタリングのタスクで高い性能を示している [Fouss 07]．これらのカーネルの有効性も自然言語処理のタスクで検証してみたい．

他に考えられる研究トピックとしては，co-training [Blum 98] のような半教師あり学習手法をグラフ理論に基づいて分析するテーマがある．本論文で示したように，ブートストラッピング法はグラフ理論に基づくアルゴリズムの一種として考えることができる．提案手法が co-training とどのように関係しているのか検討することも興味深い．

◇ 参　考　文　献 ◇

[Abney 04] Abney, S.: Understanding the Yarowsky Algorithm, *Computational Linguistics*, Vol. 30, No. 3, pp. 365–395 (2004)

[Agirre 06] Agirre, E., Martínez, D., Lacalle, de O. L., and Soroa, A.: Two graph-based algorithms for state-of-the-art WSD, in *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pp. 585–593 (2006)

[Bharat 98] Bharat, K. and Henzinger, M. R.: Improved Algorithms for Topic Distillation in a Hyperlinked Environment, in *Proceedings of the 21st ACM SIGIR Conference*, pp. 104–111 (1998)

[Blum 98] Blum, A. and Mitchell, T.: Combining Labeled and Unlabeled Data with Co-Training, in *Proceedings of the Workshop on Computational Learning Theory (COLT)*, pp. 92–100 (1998)

[Brin 98] Brin, S. and Page, L.: The Anatomy of a large-scale hypertextual Web search engine, *Computer Networks and ISDN Systems*, Vol. 30, No. 1–7, pp. 107–117 (1998)

[Chebotarev 98] Chebotarev, P. Y. and Shamis, E. V.: On Proximity Measures for Graph Vertices, *Automation and Remote Control*, Vol. 59, No. 10, pp. 1443–1459 (1998)

[Collins 99] Collins, M. and Singer, Y.: Unsupervised Models for Named Entity Classification, in *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pp. 100–110 (1999)

[Cover 76] Cover, T. M. and Hart, P. E.: Nearest neighbor pattern classification, *IEEE Transactions on Information Theory*, Vol. 13, pp. 21–27 (1976)

[Curran 07] Curran, J. R., Murphy, T., and Scholz, B.: Minimising semantic drift with Mutual Exclusion Bootstrapping, in *Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics*, pp. 172–180 (2007)

[Fouss 07] Fouss, F., Yen, L., Dupont, P., and Saerens, M.: Random-Walk Computation of Similarities between Nodes of a Graph with Application to Collaborative Recommendation, *IEEE Transactions on Knowledge and Data Engineering*, Vol. 19, No. 3, pp. 355–369 (2007)

[Hearst 92] Hearst, M.: Automatic Acquisition of Hyponyms from Large Text Corpora, in *Proceedings of the Fourteenth International Conference on Computational Linguistics*, pp. 539–545 (1992)

[Ito 05] Ito, T., Shimbo, M., Kudo, T., and Matsumoto, Y.: Application of Kernels to Link Analysis, in *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 586–592 (2005)

[Kandola 02] Kandola, J., Shawe-Taylor, J., and Cristianini, N.: Learning Semantic Similarity, in *Advances in Neural Information Processing Systems 15*, pp. 657–664 (2002)

[Kleinberg 99] Kleinberg, J.: Authoritative Sources in a Hyperlinked Environment, *Journal of the ACM*, Vol. 46, No. 5, pp. 604–632 (1999)

[Kondor 02] Kondor, R. I. and Lafferty, J.: Diffusion Kernels on Graphs and Other Discrete Input Spaces, in *Proceedings of the 19th International Conference on Machine Learning (ICML-2002)*, pp. 315–322 (2002)

[Kunegis 09] Kunegis, J. and Lommatzsch, A.: Learning Spectral Graph Transformations for Link Prediction, in *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 561–568 (2009)

[Li 08] Li, X., Wang, Y.-Y., and Acero, A.: Learning Query Intent from Regularized Click Graphs, in *Proceedings of SIGIR'08: the 31st Annual ACM SIGIR conference on Research and Development in Information Retrieval*, pp. 339–346 (2008)

[Nadler 06] Nadler, B., Lafon, S., Coifman, R., and Kevrekidis, I.: Diffusion Maps, Spectral Clustering and Eigenfunctions of Fokker-Planck Operators, *Advances in Neural Information Processing Systems 18*, pp. 955–962 (2006)

[Ng 97] Ng, H. T.: Exempler-Based Word Sense Disambiguation: Some Recent Improvements, in *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pp. 208–213 (1997)

[Ng 03] Ng, V. and Cardie, C.: Weakly Supervised Natural Language Learning Without Redundant Views, in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pp. 94–101 (2003)

[Pantel 06] Pantel, P. and Pennacchiotti, M.: Espresso: Leveraging Generic Patterns for Automatically Harvesting Semantic Relations, in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pp. 113–120 (2006)

[Riloff 99] Riloff, E. and Jones, R.: Learning Dictionaries for Information Extraction by Multi-Level Bootstrapping, in *Proceedings of the Sixteenth National Conference on Artificial Intellligence (AAAI-99)*, pp. 474–479 (1999)

[Saerens 04] Saerens, M., Fouss, F., Yen, L., and Dupont, P.: The Principal Component Analysis of a Graph, and its Relationship to Spectral Clustering, in *Proceedings of European Conference on Machine Learning (ECML 2004)*, pp. 371–383, Springer (2004)

[Smola 03] Smola, A. J. and Kondor, R. I.: Kernels and Regularization of Graphs, in *Proceedings of the 16th Annual Conference on Learning Theory*, pp. 144–158 (2003)

[Véronis 04] Véronis, J.: HyperLex: Lexical Cartography for Information Retrieval, *Computer Speech & Language*, Vol. 18, No. 3, pp. 223–252 (2004)

[Vyas 09] Vyas, V., Pantel, P., and Crestan, E.: Helping Editors Choose Better Seed Sets for Entity Set Expansion, in *Proceedings of ACM Conference on Information and Knowledge Management (CIKM-2009)*, pp. 225–234 (2009)

[Yarowsky 95] Yarowsky, D.: Unsupervised Word Sense Disambiguation Rivaling Supervised Methods, in *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pp. 189–196 (1995)

〔担当委員：高村 大也〕

2009 年 7 月 6 日 受理

著 者 紹 介

小町 守（学生会員）

2005 年東京大学教養学部基礎科学科科学史・科学哲学分科卒．2007 年奈良先端科学技術大学院大学情報科学研究科博士前期課程修了．同年後期課程に進学．修士（工学）．日本学術振興会特別研究員．大規模データを用いた意味解析に関心がある．言語処理学会第 14 回年次大会最優秀発表賞受賞．言語処理学会，情報処理学会，ACL 各会員．

工藤 拓

1976 年生．1999 年京都大学工学部電気電子工学科卒．2001 年奈良先端科学技術大学院大学情報科学研究科博士前期課程修了．2004 年同校博士後期課程修了．同年より NTT コミュニケーション科学基礎研究所リサーチアソシエイト．現在ソフトウェアエンジニアとしてグーグル株式会社に勤務．博士（工学）．統計的自然言語処理，テキストマイニング，機械学習に興味を持つ．

新保 仁（正会員）

1992 年京都大学工学部電気工学第二学科卒．1994 年京都大学大学院工学研究科修士課程電気工学第二専攻修了．1997 年京都大学大学院工学研究科博士後期課程情報工学専攻指導認定退学．博士（工学）．現在奈良先端科学技術大学院大学情報科学研究科助教．ACL，ACM，日本データベース学会各会員．

松本 裕治（正会員）

1977 年京都大学工学部情報工学科卒．1979 年同大学大学院工学研究科修士課程情報工学専攻修了．同年電子技術総合研究所入所．1984〜85 年英国インペリアルカレッジ客員研究員．1985〜87 年 (財) 新世代コンピュータ技術開発機構に出向．京都大学助教授を経て，1993 年より奈良先端科学技術大学院大学教授，現在に至る．工学博士．専門は自然言語処理．情報処理学会，日本ソフトウェア科学会，言語処理学会，認知科学会，AAAI，ACL，ACM 各会員．