平成 25 年度茨城大学工学部情報工学科卒業研究論文

領域間距離を利用した能動学習による語義曖昧性解消の領域適応

平成 26 年 2 月 7 日

茨城大学工学部情報工学科

小野寺喜行 (10T4019S)

指導教員 新納浩幸 准教授

平成 25 年度 茨城大学工学部情報工学科 卒業研究

領域間距離を利用した能動学習による語義曖昧性解消の領域適応

小野寺喜行 (10T4019S)

指導教員 新納浩幸 准教授

論文要旨

本論文では語義曖昧性解消 (Word Sense Disambiguation，WSD) の領域適応の問題に対して，データ選択に領域間距離を利用した能動学習手法を提案する．

自然言語処理のタスクにおいて帰納学習手法を用いる際，訓練データとテストデータは同じ領域のコーパスから得ていることが通常である．ただし，実際には異なる領域である場合も存在する．そこである領域（ソース領域）の訓練データから学習された分類器を，別の領域（ターゲット領域）のテストデータに合うようにチューニングすることを領域適応という．

領域適応の問題をターゲット領域のラベル付きデータの不足からくる問題として見なせば，能動学習や半教師あり学習を利用することは有効である．ここでは WSD の領域適応の問題に対して，能動学習を利用する．一般に能動学習はラベルなしデータの集合から学習効果の高いデータを選択し，そのデータにラベルを付けて訓練データに追加することで，徐々に分類器の精度を高めてゆく．能動学習のポイントはどのようにして学習効果の高いデータを選択するかである．通常は WSD の各対象単語毎に識別の信頼度を基準にしてその選択が行われる．つまり，各対象単語毎に能動学習を適用する形になっている．ただしこの形では WSD の総合的な評価であるマクロ平均やマイクロ平均の評価値を効率的に向上させることができる保証はない．そこで本論文では WSD の全対象単語の全テストデータを対象にして，ラベル付けするデータを選択することを試みる．また，選択する際の信頼度は，領域間の関係に依存していると考えられる．つまり領域間の距離が大きければ，算出した信頼度よりも更に信頼度は低く，領域間の距離が小さければ，算出した信頼度は妥当と考えられる．このように領域間距離を考慮した信頼度を用いることで更なる精度向上を図る．

実験では現代日本語書き言葉均衡コーパス（BCCWJ コーパス）における 3 つの領域 OC (Yahoo! 知恵袋)，PB (書籍) 及び PN (新聞) を利用する．SemEval-2 の日本語 WSD タスクではこれらのコーパスの一部に語義タグを付けたデータを公開しており，そのデータを利用する．すべての領域である程度の頻度が存在する多義語 16 単語を対象にして，WSD の領域適応の実験を行う．領域適応としては OC → PB，PB → PN，PN → OC，OC → PN，PN → PB，PB → OC の計 6 通りが存在し，$16 \times 6 = 96$ 通りの WSD の領域適応の問題に対して能動学習を利用した WSD の領域適応の実験を行った結果，全対象単語から選択する手法は効果があったが，領域間距離を考慮した手法には効果は見られなかった．そこで，仮に領域間距離を正しく測れているとして負の転移の有無を判定し，その判定を用いてデータ選択の手順を変える手法を試した結果，領域間距離を利用するという手法には効果が有ることが確認できた．

提案手法を効果的に利用するには，領域間距離を正しく測る方法と算出した領域間距離をラベル付けのデータ選択に取り入れる方法を考える必要がある．これらが今後の課題である．

目次

表目次

図目次

第 1 章　序論

1.1　概要

本論文では語義曖昧性解消 (Word Sense Disambiguation，WSD) の領域適応の問題に対して，データ選択に領域間距離を利用した能動学習手法を提案する．

自然言語処理のタスクにおいて帰納学習手法を用いる際，訓練データとテストデータは同じ領域のコーパスから得ていることが通常である．ただし実際には異なる領域である場合も存在する．そこである領域（ソース領域）の訓練データから学習された分類器を，別の領域（ターゲット領域）のテストデータに合うようにチューニングすることを領域適応という (Sogaard (2013))[*1]．

領域適応の問題をターゲット領域のラベル付きデータの不足からくる問題として見なせば，能動学習 (Settles (2010)) や半教師あり学習 (Chapelle et al. (2006)) を利用することは有効である．ここでは WSD の領域適応の問題に対して，能動学習を利用する．一般に能動学習はラベルなしデータの集合から学習効果の高いデータを選択し，そのデータにラベルを付けて訓練データに追加することで，徐々に分類器の精度を高めてゆく．能動学習のポイントはどのようにして学習効果の高いデータを選択するかである．通常は WSD の各対象単語毎に識別の信頼度を基準にしてその選択が行われる．つまり WSD を対象とした能動学習では，各対象単語毎に能動学習を適用する形になっている．ただしこの形では WSD の総合的な評価であるマクロ平均やマイクロ平均の評価値を効率的に向上させることができる保証はない．そこで本論文では WSD の全対象単語の全テストデータを対象にして，ラベル付けするデータを選択することを試みる．また，選択する際の信頼度は，領域間の関係に依存していると考えられる．つまり領域間の距離が大きければ，算出した信頼度よりも更に信頼度は低く，領域間の距離が小さければ，算出した信頼度は妥当と考えられる．このように領域間距離を考慮した信頼度を用いることで更なる精度向上を図る．(小野寺喜行, 新納浩幸 (2013))(小野寺喜行, 新納浩幸 (2014))

実験では現代日本語書き言葉均衡コーパス（BCCWJ コーパス (Maekawa (2007))）における 3 つの領域 OC (Yahoo! 知恵袋)，PB (書籍) 及び PN (新聞) を利用する．SemEval-2 の日本語 WSD タスク (Okumura et al. (2010)) ではこれらのコーパスの一部に語義タグを付けたデータを公開しており，そのデータを利用する．すべての領域である程度の頻度が存在する多義語 16 単語を対象にして，WSD の領域適応の実験を行う．領域適応としては OC → PB，PB → PN，PN → OC，OC → PN，PN → PB，PB → OC の計 6 通りが存在する．結果 $16 \times 6 = 96$ 通りの WSD の領域適応の問題に対して能動学習を利用した WSD の領域適応の実験を行い提案手法の有効性を示す．

[*1] 領域適応は機械学習の分野では転移学習 (神嶌敏弘 (2010)) の一種と見なされている．

1.2 本論文の構成

本論文では，最初に理論とその手法について紹介する．第 2 章では領域適応 (Domain Adaptation) について，第 3 章では語義曖昧性解消 (Word Sense Disambiguation, WSD) の概要と手法，そして WSD に良く用いられる分類器について紹介する．第 4 章では能動学習 (Active Learning) の概要や様々な手法について紹介する．

そして，提案手法とその実験結果について述べ考察を行う．第 5 章では 2 つの提案手法について述べ，第 6 章ではそれらの提案手法を用いた実験の内容とその結果を述べる．第 7 章ではその実験結果について考察を述べ，第 8 章では結論と今後の課題について述べる．

第 2 章　領域適応（Domain Adaptation）

従来，自然言語処理のタスクにおいて帰納学習手法を用いる際，訓練データとテストデータは同じ領域のコーパスから得たデータであることが通常である．しかし，あるドメインのデータのラベルを識別したいにも関わらず、別のドメインのデータからしか識別規則を学習できない場合も存在する．

そこで，ある領域の訓練データから学習された分類器を別の領域のテストデータに合うようにチューニングすることを領域適応（Domain Adaptation）という．識別規則を学習するために利用するデータのドメインをソースドメイン（ソース領域），適応される側をターゲットドメイン（ターゲット領域）と呼ぶ．例として，新聞の訓練データから学習した分類器を雑誌のテストデータに適用する流れを図 2.1 に示す．

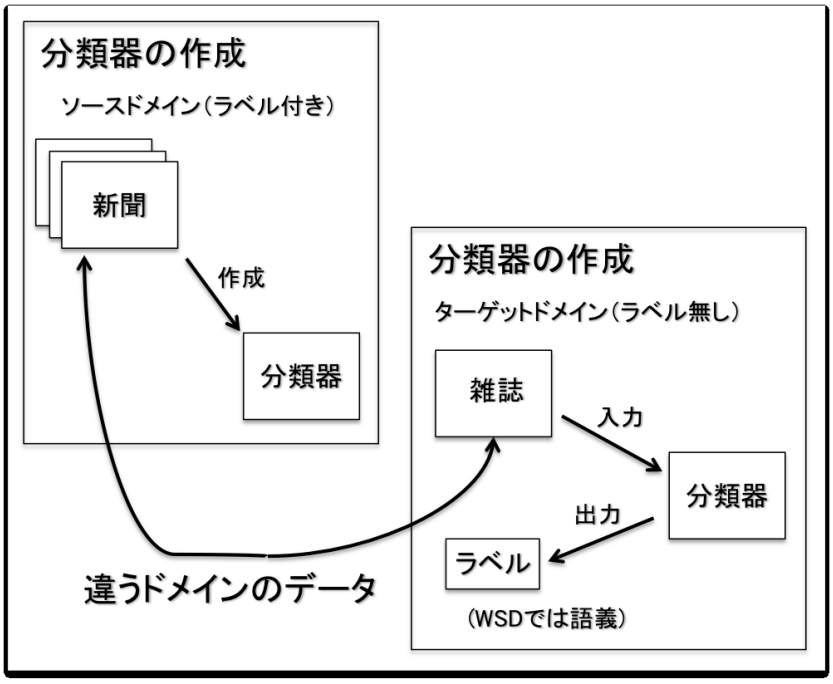

図 2.1　領域適応時の機械学習

単語には，多くの意味や正しい意味を識別する過程，または文脈間での意味が存在するので，単語のドメインが違う場合，語義の分布も異なることが多い．これは語義曖昧性解消（WSD）の領域適応の問題点である．

ソースドメインとターゲットドメインが異なる場合，同じ場合と比べて語義識別の精度が下がってしまう．これはドメイン間での語義の比率が異なることによって起こる問題である．この精度を向上させることが領域適応の研究における目的となっている．

第 3 章　語義曖昧性解消 (WSD)

3.1　語義曖昧性解消 (WSD) について

3.1.1　概要

単語には，一般に複数の意味が存在する．語義曖昧性解消 (Word Sense Disambiguation, WSD) とはそのような多義語の語義を識別し，正しい語義を割り当てる処理のことである．例えば「与える」という単語には表 3.1 に示すような語義があり，WSD ではこのような多義語を対象として語義曖昧性を解消するための処理を行う．

表 3.1　「与える」の語義

単語	語義
与える	(1)　自分の物を他人に渡し，その人のものとする．やる． ▽ 現在では上の者が恩恵的な意味で授ける場合に使う． (2)　あてがう (1).「課題を—」「ヒントを—」 (3)　こうむらせる．「損害を—」「不安を—」 ▽「当たる」「値」と同じ語源.

3.1.2　一般的な手法

WSD には、教師有り学習 (supervised learning)，半教師有り学習 (semi-supervised learning)，教師無し学習 (unsupervised learning) などが存在する．

教師有り学習は，ソースドメインからの十分な量のラベル付きデータとターゲットドメインの少量のラベル付きデータを用いて学習を行う手法である．半教師有り学習は，ソースドメインからの十分な量のラベル付きデータとターゲットドメインの十分な量のラベル無しデータを用いて学習を行うものである．これは，ラベル有りとラベル無しが混在するデータから学習することで，ラベルありのデータだけで学習したときよりも精度を上げることを目的としている．教師無し学習は，ソースドメインのラベル付きデータのみで学習する手法である．例として，教師有り学習による WSD の流れを図 3.1 に示す．

学習には，文書から大量に集めたコーパスから得られるデータを利用する．このコーパスには，単純に文書を集めたコーパスのほかに，文書中の単語に語義の情報や構文構造などの付加情報を含む自然言語処理に特化した注釈つきのコーパスが存在する．その注釈付きのコーパスを使うことで高い語義識別の精度を得ることが出来，多くのデータを使うことで精度の向上が期待できるが，大量の文書に付加情報を付けていく作業は手間がかかる上に相当な時間とコストを必要とする．学習データとなるものには文書中の単語に品詞をつけた POS タグや，対象単語の文脈に含まれる単語を集めたもの (bag-of-words) などがある．

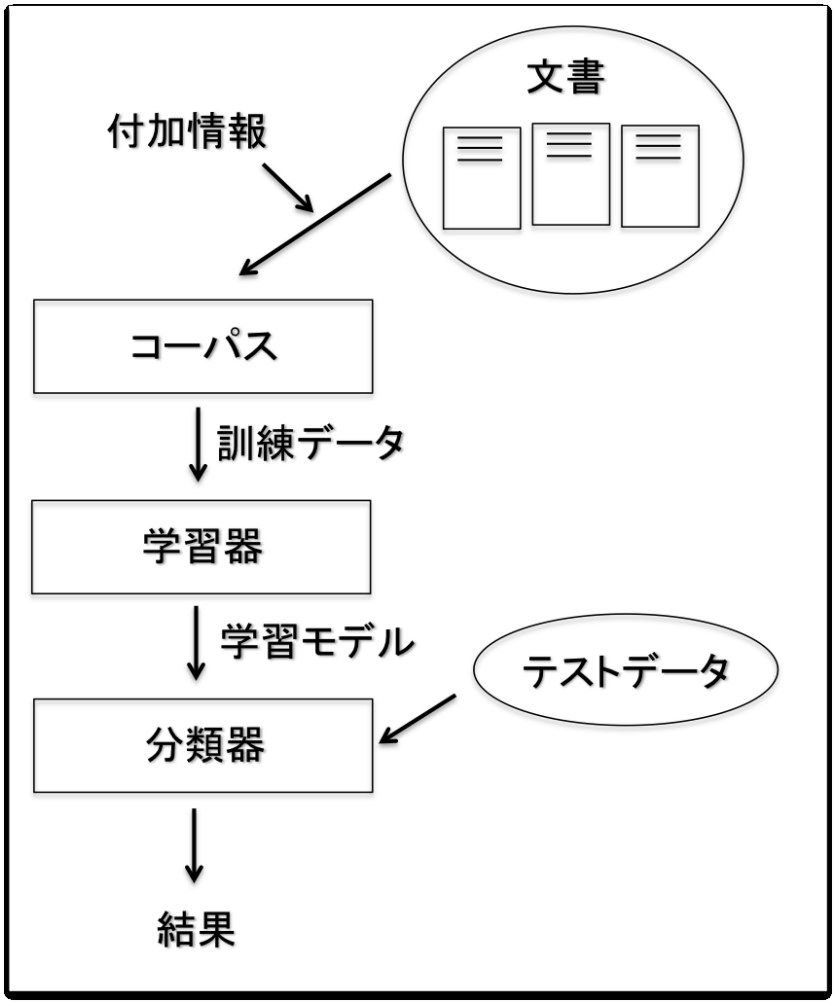

図 3.1　WSD における教師あり学習の流れ

3.2　サポートベクタマシン（SVM）

3.2.1　概要

本研究では，学習や識別においてサポートベクタマシン (Support Vector Machine, SVM) を用いる．SVM は自然言語処理の分野でよく使用される線形二値分類器であり，非常に高い分類性能を持つことが知られている．また，カーネル法を用いれば SVM は非線形な分類も可能である．

SVM は線形二値分類器でありクラス（データを分ける予め決められたグループ）の数が 2 つであるような問題に用いられてきた．2 つのクラスはそれぞれ「正クラス」及び「負クラス」と呼ばれ，正クラスに属する事例は正例、負クラスに属する事例は負例と呼ばれる．

ここで，訓練データ D が以下で与えられるとする．

$$D = \{\left(\mathbf{x}^{(1)}, y^{(1)}\right), \left(\mathbf{x}^{(2)}, y^{(2)}\right), \cdots, \left(\mathbf{x}^{(|D|)}, y^{(|D|)}\right)\}$$

$x^{(1)}, x^{(2)}, \cdots, x^{(|D|)}$ は事例の素性ベクトルであり $y^{(1)}, y^{(2)}, \cdots, y^{(|D|)}$ はそれぞれの事例のクラスラベルである．正例のクラスラベルは $+1$，負例のクラスラベルは-1 である．SVM は線形分類器であるの

で，分離平面の方向ベクトル $\mathbf{w}$ と切片 b をパラメータとして

$$f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} - b$$

と表される．

事例 $\mathbf{x}$ は $f(\mathbf{x}) \geq 0$ ならば正クラス，$f(\mathbf{x}) \leq 0$ ならば負クラスに分類される．

3.2.2 マージン最大化

ここでは，パラメータ $\mathbf{w}$ と b を求めるための基本的な考え方を説明する．訓練データが図 3.2 のように分布していると仮定する．

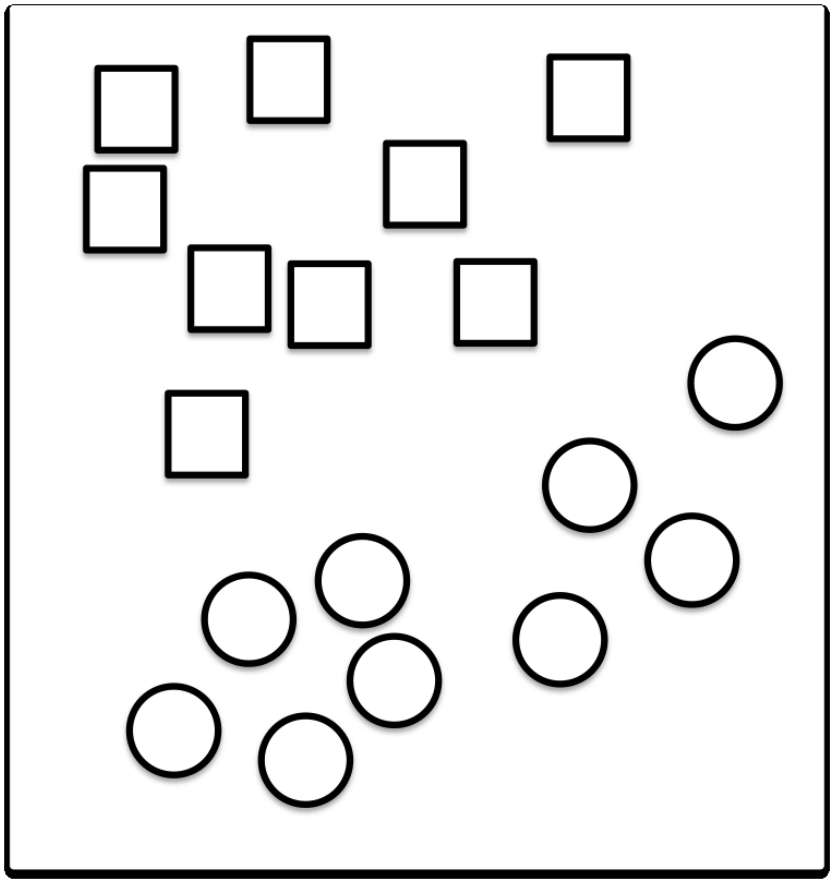

図 3.2　訓練データの分布

これを分類する平面を分類平面と呼ぶ．SVM の目的は良い分類平面を構築することである．方向ベクトルを $\mathbf{w}$，切片を b で表すと分類平面は $\mathbf{w} \cdot \mathbf{x} = b$ を満たす点 $\mathbf{x}$ の集合となる．

図 3.2 を見ると，この訓練データを分類できる分類できる分類平面は多数存在することが分かる．SVM では図 3.3 にある様な分類平面が「どちらのクラスからもなるべく遠い位置で分ける」という戦略が，マージン最大化とよばれる戦略である．分類平面のマージンとは最も近い訓練事例への距離として定義される．

分離平面の最も近くにある正例を $\mathbf{x}_+$ とし，$\mathbf{x}_+$ と分離平面を結ぶ垂線の足を $\mathbf{x}_*$ で表すとする．ここで、マージンは $|\mathbf{x}_+ - \mathbf{x}_*|$ と表せられる．$\mathbf{w}$ と $\mathbf{x}_+ - \mathbf{x}_*$ は同じ方向を向いているので，

$$\mathbf{w} \cdot (\mathbf{x}_+ - \mathbf{x}_*) = |\mathbf{w}| \, |\mathbf{x}_+ - \mathbf{x}_*|$$

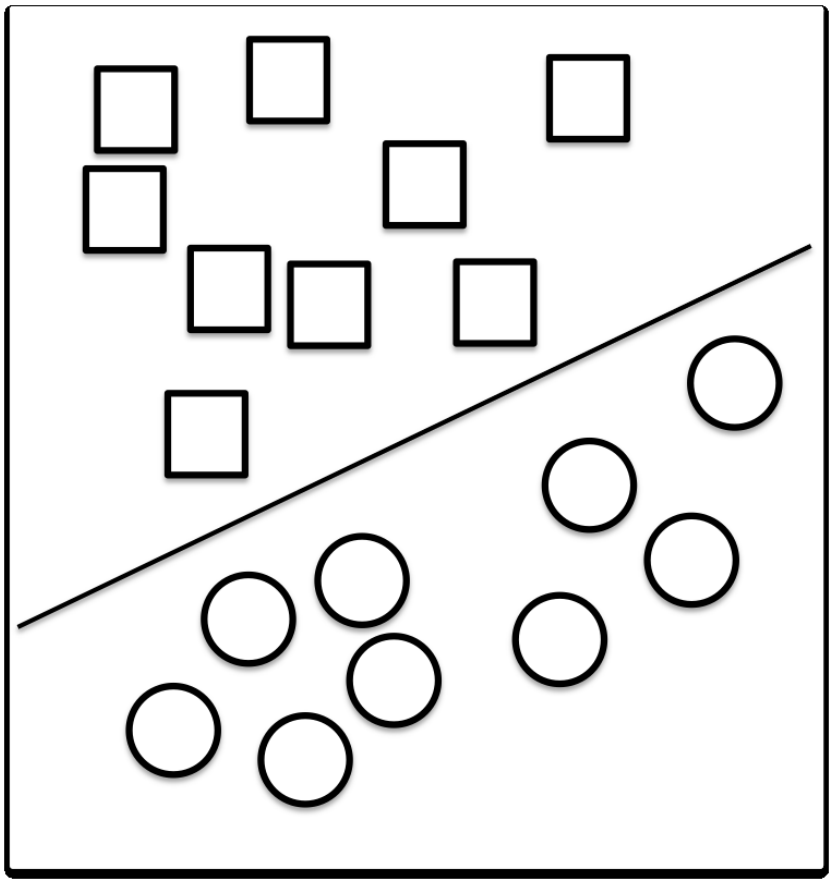

図 3.3　分離平面の例

が成り立つ.

分離平面 $\mathbf{w}\cdot\mathbf{x}=b$ は式全体を定数倍しても変わらないので，うまく定数倍すれば $\mathbf{w}\cdot\mathbf{x}-b=1$ とすることが出来る．このようにパラメータが調整されているとする．また，$\mathbf{x}_*$ は分離平面上にあるので当然ながら $\mathbf{w}\cdot\mathbf{x}_*=b$ である．よって，

$$\begin{aligned}\mathbf{x}_+-\mathbf{x}_* &= \mathbf{w}\cdot\mathbf{x}_+-\mathbf{w}\cdot\mathbf{x}_*\\ &= (b+1)-b\\ &= 1\end{aligned}$$

となる.

$\mathbf{w}\cdot(\mathbf{x}_+-\mathbf{x}_*)=|\mathbf{w}|\,|\mathbf{x}_+-\mathbf{x}_*|$ と合わせると $|\mathbf{w}|\,|\mathbf{x}_+-\mathbf{x}_*|=1$ であることがわかるので,

$$|\mathbf{x}_+-\mathbf{x}_*|=\frac{1}{|\mathbf{w}|}$$

が導ける.

$|\mathbf{x}_+-\mathbf{x}_*|$ はマージンを表しているので，k の分離平面のマージンは

$$\frac{1}{|\mathbf{w}|}$$

で表される．このままでは扱いづらいため，2 乗して

$$\frac{1}{\mathbf{w^2}}$$

を最大化する．さらに，分数では扱いづらい為この逆数を最小化することにする．つまり，$\mathbf{w^2}$ を最小化することになる．

3.2.3 厳密制約化の SVM のモデル

当然ながら訓練事例は正しく分類される必要がある．$y^{(i)} = +1$ であるような訓練事例については $\mathbf{w} \cdot \mathbf{x}^{(i)} - b \geq 1$ であれば良く，$y^{(i)} = -1$ であるような訓練事例については $\mathbf{w} \cdot \mathbf{x}^{(i)} - b \leq 1$ であれば良い．この条件は次のようにまとめることができる．

$$y^{(i)} \left(\mathbf{w} \cdot \mathbf{x}^{(i)} - b \right) \geq 1$$

よって，これを制約とした次の最適化問題を解けばよい．

$$\begin{aligned} min.\ \ & \frac{1}{2}\mathbf{w}^2 \\ s.t.\ \ & y^{(i)} \left(\mathbf{w} \cdot \mathbf{x}^{(i)} - b \right) - 1 \geq 0; \forall i \end{aligned}$$

ここで，目的関数につけた係数 $1/2$ は計算をわかりやすくするためであり，省略しても構わない．

この不等式制約付き最適化問題は凸計画化問題であるので，ラグランジュ法を用いて解く事が可能である．ラグランジュ未定乗数 α_i を導入すると，ラグランジュ関数は

$$L\left(\mathbf{w}, b, \alpha\right) = \frac{1}{2}\mathbf{w}^2 - \sum_i \alpha_i \left(y^{(i)} \left(\mathbf{w} \cdot \mathbf{w}^{(i)} - b \right) - 1 \right)$$

と表せる．これをそれぞれのパラメータで偏微分すると次のようになる．

$$\begin{aligned} \nabla \mathbf{w} L\left(\mathbf{w}, b, \alpha\right) &= \mathbf{w} - \sum_i \alpha_i y^{(i)} \mathbf{w}^{(i)} \\ \frac{\delta L\left(\mathbf{w}, b, \alpha\right)}{\delta b} &= \sum_i \alpha_i y^{(i)} \end{aligned}$$

これらを 0 とおいて，

$$\begin{aligned} \mathbf{w}^* &= \mathbf{w} - \sum_i \alpha_i y^{(i)} \mathbf{x}^{(i)} \\ \sum_i \alpha_i y^{(i)} &= 0 \end{aligned}$$

を得る．1 つ目の式にある分離平面の方向ベクトル $\mathbf{w}^*$ は訓練事例ベクトルの線形和で表されることを意味する．この式を分離平面の式 $\mathbf{w} \cdot \mathbf{x} = b$ に代入すると，

$$f\left(\mathbf{x}\right) = \sum_i \alpha_i y^{(i)} \mathbf{x}^{(i)} \cdot \mathbf{x} - b$$

となる．この式で α_i と b を求めることが出来れば，分離平面を得ることが可能である．

これらの式を元のラグランジュ関数に代入すると．

$$L\left(\mathbf{w}^*, b, \alpha\right) = -\frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y^{(i)} y^{(j)} x^{(j)} \cdot x^{(i)} + \sum_i \alpha^i$$

となる．これで元々の変数 $\mathbf{w}$ と b がラグランジュ関数から消去された．

ラグランジュ関数 $L(\mathbf{x}, \lambda)$ において，次の不等式

$$L(\mathbf{x}^*, \lambda^*) \geq L(\mathbf{x}, \lambda)$$

を満たす点 $L(\mathbf{x}^*, \lambda^*)$ は鞍点と呼ばれる．この点は $\mathbf{x}$ に関して最大となり、λ に関して最小となる．

今回のラグランジュ関数に置き換えると，$L(\mathbf{w}^*, b, \alpha)$ を最大化する α_i を求めれば良い．ただし，$\sum_i \alpha_i y^{(i)} = 0, \alpha_i \leq 0$ の制約下での最大化である．

最適解 α_i^* が求まれば $\mathbf{w}^*$ が求まり，$\mathbf{x}_+$ を用いて $b = \mathbf{x}^* \cdot \mathbf{x}_+ - 1$ として切片も求まる．

3.2.4 緩和制約化の SVM のモデル

厳密制約化の SVM のモデルは実際のデータに対してはなかなか上手く分類できない．それは，すべての訓練事例が正確に分類されなくてはならないという制約 $y^{(i)}\left(\mathbf{w} \cdot \mathbf{x}^{(i)} - b\right) - 1 \geq 0$ があるためである．訓練データの中には例外的な事例が存在する事が多く，そのような事例によって分類平面が大きく影響を受けてしまう．極端なケースでは、訓練データが線形関数でうまく分類できずに，制約を満たす解が存在しないことにもなってしまう．

そこで，制約を少し緩めることを考える．新たな変数 ξ_i を導入し

$$y^{(i)}\left(\mathbf{w} \cdot \mathbf{x}^{(i)} - b\right) \geq -\xi_i$$

という制約に書き換える．ξ_i は i 番目の訓練事例がうまく分けられない度合いを表す．つまり，ξ_i は小さいほうが良い．そこで，これを目的関数に加えると新しい最適化問題は次のようになる．

$$\begin{aligned} min.\ \ & \frac{1}{2}\mathbf{w}^2 + C\sum_i \xi_i \\ s.t.\ \ & y^{(i)}\left(\mathbf{w} \cdot \mathbf{x}^{(i)} - b\right) \geq 1 - \xi_i; \forall i,\ \xi_i \geq 0; \forall i \end{aligned}$$

ここで C は正の定数であり，この値が大きいほど正確に分類できるようになる．

この最適化問題をラグランジュ法を用いて解く．ラグランジュ未定乗数 α_i を導入すると，ラグランジュ関数は

$$L(\mathbf{w}, b, \xi, \alpha, \beta) = \frac{1}{2}\mathbf{w}^2 + C\sum_i \xi_i - \sum_i \alpha\left(y^{(i)}\left(\mathbf{w}\mathbf{x}^{(i)} - b\right) - 1 + \xi_i\right) - \sum_i \beta_i \xi_i$$

と表せる．これをそれぞれのパラメータで偏微分しそれぞれを 0 とすると $\mathbf{w}$ と b については厳密制約下の場合と同じで

$$\mathbf{w}^* = \mathbf{w} - \sum_i \alpha_i y^{(i)} \mathbf{x}^{(i)}$$

$$\sum_i \alpha_i y^{(i)} = 0$$

が得られる．ξ_i に関しては，

$$\frac{\delta L(\mathbf{w}, b, \xi, \alpha, \beta)}{\delta \xi_i} = C - \alpha_i - \beta_i$$

であるので，0 とすると

$$C = \alpha_i + \beta_i$$

を得る．これらの等式を合わせて得られる双対ラグランジュ関数は次のようになる．

$$L\left(\mathbf{w}^*, b, \alpha, \beta\right) = -\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y^{(i)}y^{(j)}x^{(i)}\cdot x^{(j)} + \sum_i \alpha_i$$

ただし，$0 \leq \alpha \leq C$ の制約のもとで最大化する．

b は $0 \leq \alpha_i \leq C$ となる事例を 1 つ持ってきて

$$b = \mathbf{w}\cdot\mathbf{x}^{(i)} - y^{(i)}$$

とすることで計算できる．

第 4 章　能動学習 (Active Learning)

4.1　概要

精度の高い分類器を構築するためにはラベル付きデータを増やせばよいが，ラベル付けは高コストという問題がある．例えば，発話文のラベル付けにかかる時間が実時間長の 10 倍以上である．固有表現や関係抽出のためのラベル付けでは平易なニュース 1 つでも 1 時間半以上かかる上，専門知識も必要である．また，言語判定のために 19 言語 70 万件のツイートに言語ラベルを付与する作業は 5 ヶ月を要する．そのため少量のラベル付きデータで分類器の精度を上げることが望まれ，能動学習 (Active Learning) はそのような背景から考案された学習手法である．

能動学習では，まず正解無しデータの中から「このデータの正解が分かれば性能が上がるだろう」という学習効果の高いデータを選択する．そのデータを Oracle(信託：正解を教えてくれる何か) にデータを問い合わせ (query)，得た正解をラベル付きデータに追加する．この例を図 4.1 に示す．

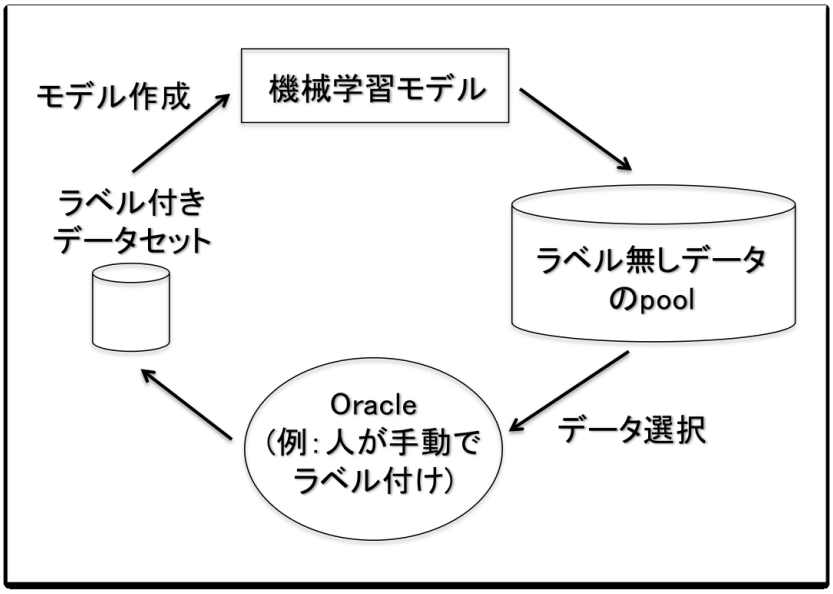

図 4.1　能動学習の例

4.2　分類

能動学習には，ラベル付けするデータを何処から得るかについて次の 3 種類の方法がある．

- membershipquery synthesis

—正解を知りたいデータを新規に生成する.

- stream-based selective sampling
  —ストリームで流れてくるデータ 1 つ 1 つについて Oracle に問い合わせるかどうか判断する.
- pool-based sampling
  —ラベルなしデータの中から次に問い合わせるデータを 1 つ選ぶ.

また，ラベル付けするデータを選ぶ方法には次の 6 種類の戦略がある.

1. Uncertainly Sampling
2. Query-By-Committee
3. Expected Model Change
4. Expected Error Reduction
5. Variance Reduction
6. Density-Weighted Methods

Uncertainly Sampling は「最も不確かなデータ」を選択する戦略であり，不確かさの測り方には次に示す様な方法がある.

- Least Confident
  —「確率最大のラベル」の確率が最小のデータを選択

$$\arg\min_x \left(\max_y P\left(y|x\right) \right)$$

- Margin Sampling
  —(「1 番目に確率の高いラベル」の確率 −「2 番目に確率の高いラベル」の確率）が最小のデータを選択

$$\arg\min_x \left(P\left(y_1|x\right) - P\left(y_2|x\right) \right)$$

- Entropy-based Approach
  —予測分布のエントロピーが最大のデータを選択

$$\arg\max_x \left(-\sum_y P\left(y|x\right) \log P\left(y|x\right) \right)$$

Query-By-Committee は「多数決で表が 1 番割れたデータ」を選ぶ戦略で，不確かなデータを選ぶ点では Uncertainly sampling と同様である．アンサンブル（複数の分類器の結果を利用する）を前提としており精度の嵩上げも期待できる．具体的なデータ選択の方法には次に示す様なものがある.

- Vote Entropy
  —投票結果情報のエントロピーを計算して，それが最も大きいものを選択する．各分類器は確率モデルでなくても良い.

$$\arg\max_x \left(-\sum_y \frac{V\left(y\right)}{C} \log \frac{V\left(y\right)}{C} \right)$$

(C:分類器数, $V\left(y\right)$:ラベル y を予測した分類器数)

- Average Kullback-Leibler Divergence
  —複数の学習器の予測分布それぞれと，それらを平均した分布との KL-divergence の平均が大きいものを選択する．つまり，各分類器の出力する予測分布動詞が最も似ていないデータを選択する．

Uncertainly Sampling も Query-By-Committee も各データ点ごとの評価しか行わないが，学習して効果があるかどうかは分布にもよるはずである．Density-Weighted Methods では次に示す方法の様なデータ点の分布も考慮した評価をする戦略である．

- Information Density
  —評価に類似度を加味した係数を掛ける．「似ているデータ」が多いほど選ばれやすい．

$$\arg\max_{x} \phi_A(x) \left(\frac{1}{U} \sum_{u=1}^{U} \mathrm{sim}(x, x_u) \right)^{\beta}$$

($\phi_A(x)$:手法 A におけるデータ x の評価関数, U:pool サイズ, x_u:pool 内の u 番目の点)

Expected Error Reduction は未知である正解とテストデータについてテストデータの誤差を期待誤差 (risk) に，正解ラベルを推定値もしくは期待値に置き換える戦略である．$P(y|x;L)$ を訓練データ L で学習したモデルにおいて，点 x におけるラベル y の確率とすると訓練データに (x_i, y_i) を追加した場合の log-loss は次の式で得られる．

$$-\sum_{x \in U} \sum_{y} P(y|x, L \cup \{(x_i, y_i)\}) \log P(y|x, L \cup \{(x_i, y_i)\})$$

これを $R(x_i, y_i; L)$ と書く．y_i が未知の為，現在のモデルでの期待値を評価値とする．

$$E_{y_i}[R(x_i, y_i; L)] = \sum_{y_i} P(y_i|x_i; L) R(x_i, y_i; L)$$

これを最小とするような x_i を query とする．「学習後のエントロピーの総和を最小化」と同値である．(中谷秀洋 (2013))

4.3 Schohn の手法

先に示した様に能動学習には様々な手法が存在するが，本論文では簡易でありながら効果が高い Schohn の手法 (Schohn and Cohn (2000)) を通常の手法として利用する．Schohn の手法を図 4.2 に示す．

(1)	ラベル付きデータから分類器を作成する．
(2)	作成した分類器によりラベルなしデータを識別する．このとき識別の信頼度も求める．
(3)	各対象単語で識別の信頼度が最も低いデータにラベルを付け，ラベル付きデータに追加する．
(4)	(1) に戻る

図 4.2　能動学習の手順 (Schohn の手法)

第 5 章　提案手法

5.1　全対象単語に対する能動学習

通常，能動学習では Schohn の手法の様に，WSD の各対象単語毎に識別の信頼度を基準にしてその選択が行われる．つまり WSD を対象とした能動学習では各対象単語毎に能動学習を適用する形になっているが，この形では WSD の総合的な評価であるマクロ平均やマイクロ平均の評価値を効率的に向上させることができる保証はない．

そこで WSD の全対象単語の全テストデータを対象にして，その中から信頼度が最も低いデータをラベル付けするデータとして選択することを試みる．手順は図 5.1 に示す通りである．

(1)	ラベル付きデータから分類器を作成する．
(2)	作成した分類器によりラベルなしデータを識別する．このとき識別の信頼度も求める．
(3)	全ての対象単語の中で識別の信頼度が最も低いデータにラベルを付け，ラベル付きデータに追加する．
(4)	(1) に戻る

図 5.1　全対象単語に対する能動学習

5.2　領域間距離を用いた能動学習

ラベル付けするデータを選択する際の信頼度は，領域間の関係に依存していると考えられる．つまり領域間の距離が大きければ算出した信頼度よりも更に信頼度は低く，領域間の距離が小さければ算出した信頼度は妥当と考えられる．そこで，領域間距離を考慮した信頼度を用いることで更なる精度向上を図る．

ここでは，確率密度比から求めた重みを領域間類似度と考え領域間距離とし，能動学習においてラベル付けするデータの選択に識別の信頼度と領域間距離から導き出した評価値を用いることにする．WSD の全対象単語の全テストデータを対象にして次に示すように評価値を求め，図 5.2 に示す手順で学習を行う．

まず，分類器によって識別した n 個のラベルなしデータ

$$x_j \, (j = 1, 2, \ldots, n)$$

に対する識別の信頼度の集合

$$h_j \, (j = 1, 2, \ldots, n)$$

の平均値 m と標本分散値 u^2 を以下の式で求める．

$$m = \frac{\sum_{j=1}^{n} h_j}{n}$$
$$u^2 = \frac{\sum_{j=1}^{n} (m - h_j)^2}{(n-1)}$$

次に，領域間類似度の集合

$$s_j \, (j = 1, 2, \ldots, n)$$

から同様にして平均値 m' と標本分散値 u'^2 を求める．

$$m' = \frac{\sum_{j=1}^{n} s_j}{n}$$
$$u'^2 = \frac{\sum_{j=1}^{n} (m - s_j)^2}{(n-1)}$$

そして各データ x_j について以下の値を求める．

$$r_j = \frac{(h_j - m)}{u}$$
$$w_j = \frac{u \times (s_j - m')}{u'} + m$$

この r_j と w_j の値を合計したものがデータ x_j の評価値となる．

(1)	ラベル付きデータから分類器を作成する．
(2)	作成した分類器によりラベルなしデータを識別する．このとき識別の信頼度も求める．
(3)	識別の信頼度と領域間距離から各データの評価値を求める．
(4)	全ての対象単語の中で評価値が最も低いデータにラベルを付け，ラベル付きデータに追加する．
(5)	(1) に戻る

図 5.2　領域間距離を用いた能動学習

第 6 章　実験

6.1　実験概要

6.1.1　実験内容

実験では現代日本語書き言葉均衡コーパス（BCCWJ コーパス (Maekawa (2007)))における 3 つの領域 OC (Yahoo! 知恵袋)，PB (書籍) 及び PN (新聞) を利用する．SemEval-2 の日本語 WSD タスク (Okumura et al. (2010)) ではこれらのコーパスの一部に語義タグを付けたデータを公開しており，そのデータを利用する．すべての領域である程度の頻度が存在する多義語 16 単語 (表 6.1) を対象にして，WSD の領域適応の実験を行う．

表 6.1　実験対象単語

単語	辞書上の語義数	OC		PB		PN	
		頻度	語義数	頻度	語義数	頻度	語義数
言う	3	666	2	1114	2	363	2
入れる	3	73	2	56	3	32	2
書く	2	99	2	62	2	27	2
聞く	3	124	2	123	2	52	2
子供	2	77	2	93	2	29	2
時間	4	53	2	74	2	59	2
自分	2	128	2	308	2	71	2
出る	3	131	3	152	3	89	3
取る	8	61	7	81	7	43	7
場合	2	126	2	137	2	73	2
入る	3	68	4	118	4	65	3
前	3	105	3	160	2	106	4
見る	6	262	5	273	6	87	3
持つ	4	62	4	153	3	59	3
やる	5	117	3	156	4	27	2
ゆく	2	219	2	133	2	27	2
平均	3.35	193.9	2.94	150.6	2.88	75.56	2.69

領域適応としては OC → PB，PB → PN，PN → OC，OC → PN，PN → PB，PB → OC の計 6 通りが存在する (表 6.2)．結果 $16 \times 6 = 96$ 通りの WSD の領域適応の問題に対して通常の能動学習と 2 つの提案手法，「全対象単語に対する能動学習」と「領域間距離を用いた能動学習」による識別の平均正解率で比較する．なお，ラベル付けするデータの数は従来手法は各単語 5 点ずつ計 80 点，提案手法は全体で 80 点とした．加えて，ランダムにラベル付けするデータを選択する場合も試す．こちらは各単語 5 点ずつ選

表 6.2　領域適応の組み合わせ

領域適応のタイプ	ソースドメイン	ターゲットドメイン
OC → PB	Yahoo!知恵袋	書籍
PB → PN	書籍	新聞
PN → OC	新聞	Yahoo!知恵袋
OC → PN	Yahoo!知恵袋	新聞
PN → PB	新聞	書籍
PB → OC	書籍	Yahoo!知恵袋

択する場合と全体から 80 点選択する場合それぞれ 5 回行いその平均を求める．分類器は SVM を用い，SVM ツールには libsvm[*1] を使用した．libsvm では `-b` オプションにより識別の信頼度が求められる．

6.1.2　マクロ平均／マイクロ平均

この実験ではマクロ平均とマイクロ平均という 2 通りの正解率の平均を算出し評価を行う．

マクロ平均とは，セット数 N のテストをする場合，各セットの平均値を計算した後それらの値を平均するものである．この実験では以下に示すようにマクロ平均を算出する．なお，W は実験を行う単語の集合，n_i は単語 i の分類したデータの数，x_i は単語 i の分類での正解数を表す．

$$\frac{1}{16}\sum_{i \in W}\frac{x_i}{n_i}$$

また，マイクロ平均とは，セット数 N のテストをする場合，全テストの値を合計してから，平均値を計算するものである．この実験では以下に示すようにマイクロ平均を算出する．

$$\frac{\sum_{i \in W} x_i}{\sum_{i \in W} n_i}$$

6.2　実験結果

実験結果を表 6.3(マクロ平均) と表 6.4(マイクロ平均) に示す．結果として，「全対象単語に対する能動学習」は通常の手法よりも主にマクロ平均において良い正解率となり識別精度の向上に効果が有ると言える．また，「領域間距離を用いた能動学習」は低い正解率となり精度の向上に効果は見られなかった．

[*1] `http://www.csie.ntu.edu.tw/ cjlin/libsvm/`

表 6.3　実験結果 (マクロ平均 (%))

	通常の能動学習	ランダムに選択 (各単語 5 点ずつ)	ランダムに選択 (全体から 80 点)	全対象単語に対する能動学習	領域間距離を用いた能動学習
OC → PB	74.80	73.55	72.40	75.49	71.76
PB → PN	79.74	72.29	72.40	81.14	76.94
PN → OC	73.39	71.23	69.89	74.38	68.43
OC → PN	75.23	71.59	69.33	73.86	70.47
PN → PB	78.26	76.40	72.09	79.55	77.33
PB → OC	76.22	74.30	72.09	77.03	72.04

表 6.4　実験結果 (マイクロ平均 (%))

	通常の能動学習	ランダムに選択 (各単語 5 点ずつ)	ランダムに選択 (全体から 80 点)	全対象単語に対する能動学習	領域間距離を用いた能動学習
OC → PB	78.89	77.85	78.33	78.45	77.36
PB → PN	83.37	82.47	78.33	84.62	83.13
PN → OC	74.82	73.94	74.85	74.99	72.88
OC → PN	75.60	74.80	77.49	67.00	65.26
PN → PB	82.09	81.02	74.69	81.65	83.03
PB → OC	78.70	76.35	74.69	78.20	74.57

第 7 章　考察

能動学習において領域間距離の利用は効果が無いのか考察する．仮に領域間距離を正しく測れているとすると，負の転移の有無を判定できる．その負の転移が生じているかどうかを調べるために以下の実験を行った．

単語 w_i についてソース領域 S からターゲット領域 T への領域適応の実験を行う．まずターゲット領域 T のラベル付きデータをランダムに 15 個取り出し，残りを評価データとする．つまり利用できる訓練データはソース領域 S のラベル付きデータとターゲット領域 T からランダムに取り出した 15 個のラベル付きデータとなる．この訓練データを用いて手法 A により分類器を作成し，先の評価データの語義識別の正解率 $P_{i,k}$ を測る．この実験を 5 回行い $P_{i,1}, P_{i,2}, \cdots, P_{i,5}$ を得る．それらの平均 P_i を「単語 w_i の S から T への領域適応における手法 A の平均正解率」とする．上記の単語 w_i を 16 種類の各対象単語 $w_1, w_2, \cdots, w_{16}$ に変えることで，16 個の平均正解率 $P_1, P_2, \cdots, P_{16}$ が得られる．それらの平均 P を「S から T への領域適応における手法 A の平均正解率」とする．

上記の手法 A としては，以下の 3 種類を試す．(1) ソース領域のラベル付きデータのみを用いる手法 (ターゲット領域の 15 個のラベル付きデータの重みを 0 とする手法) (S-Only)，(2) ターゲット領域からランダムに取り出した 15 個のラベル付きデータのみを用いる手法 (ソース領域のラベル付きデータの重みを 0 とする手法) (T-Only)，(3) ソース領域のラベル付きデータとターゲット領域の 15 個のラベル付きデータを用いる手法 (S+T)．

S から T への領域適応における各手法の平均正解率を表 7.1 に示す．

負の転移が生じているかどうかの判定には，上記の実験でより得られた `T-Only`，`S-Only` 及び `S+T` の正解率を利用する．もしも正解率で以下の関係が成立しているなら，負の転移が生じていないと考えられる．

```
T-Only, S-Only  <  S+T
```

結果を表 7.2 に示す．チェックがつけられた箇所が負の転移が生じていない領域適応の問題である．96 種類の領域適応の問題の中で 44 種類において負の転移が生じていない．

この表をもとに，負の転移が生じていない a 個の単語については図 7.1 に示す手順で従来手法と同様の手順で各単語 5 点ずつラベル付きデータを追加し，負の転移が生じている 16-a 個の単語は全対象単語に対する能動学習の様な図 7.2 に示す手順でその中で信頼度最低のものをラベル付きデータに追加を計 (16-A)×5 回行う手法を試す．

実験結果を表 7.3(マクロ平均) と表 7.4(マイクロ平均) に示す．通常の能動学習よりも良い結果を得られ，「全対象単語に対する能動学習」よりもマイクロ平均において良い値となった．よって，領域間距離の情報を用いてラベル付けするデータを選択する手順を制御することは効果が有ると言える．

提案手法の平均正解率が低い原因は領域間距離を利用すること自体ではなく，「確率密度比から領域間

(1) ラベル付きデータから分類器を作成する．
(2) 作成した分類器によりラベルなしデータを識別する．このとき識別の信頼度も求める．
(3) 負の転移が生じていない各単語で識別の信頼度が最も低いデータにラベルを付け，ラベル付きデータに追加する．
(4) (1) に戻る

図 7.1　負の転移が生じていない単語の学習の手順

(1) ラベル付きデータから分類器を作成する．
(2) 作成した分類器によりラベルなしデータを識別する．このとき識別の信頼度も求める．
(3) 負の転移が生じている全ての単語の中で識別の信頼度が最も低いデータにラベルを付け，ラベル付きデータに追加する．
(4) (1) に戻る

図 7.2　負の転移が生じている学習の手順

距離を測る」という方法が正しく領域間距離を測ることが出来なかった事や，領域間距離をラベル付けするデータ選択に上手く取り入れられなかった事が考えられる．したがって，今後は領域間距離を正しく測る方法と領域間距離をラベル付けするデータ選択に取り入れる方法を考える必要がある．

表 7.1　各手法の平均正解率

領域適応	S-Only	T-only	S+T
OC → PB	0.7137	0.7559	0.7511
PB → PN	0.7678	0.7206	0.7801
PN → OC	0.6926	0.7716	0.7630
OC → PN	0.6829	0.7300	0.7324
PN → PB	0.7543	0.7561	0.7863
PB → OC	0.6988	0.7766	0.7533
平均	0.7184	0.7518	0.7611

表 7.2　負の転移が生じていない領域適応

単語	OC → PB	PB → PN	PN → OC	OC → PN	PN → PB	PB → OC
言う		✓	✓	✓	✓	
入れる		✓	✓	✓	✓	✓
書く	✓			✓	✓	
聞く	✓					
子供			✓		✓	
時間	✓		✓		✓	
自分	✓	✓				
出る				✓		✓
取る			✓		✓	✓
場合		✓	✓		✓	✓
入る		✓	✓		✓	✓
前		✓				
見る	✓					
持つ	✓	✓				✓
やる		✓		✓		✓
ゆく		✓		✓	✓	

表 7.3　実験結果 (マクロ平均 (%))

	通常の能動学習	ランダム (各単語 5 点)	ランダム (全体から)	全対象単語に対する	領域間距離を利用	負の転移を考慮
OC → PB	74.80	73.55	72.40	75.49	71.76	75.92
PB → PN	79.74	72.29	72.40	81.14	76.94	80.71
PN → OC	73.39	71.23	69.89	74.38	68.43	74.17
OC → PN	75.23	71.59	69.33	73.86	70.47	75.01
PN → PB	78.26	76.40	72.09	79.55	77.33	78.08
PB → OC	76.22	74.30	72.09	77.03	72.04	75.66

表 7.4　実験結果 (マイクロ平均 (%))

	通常の能動学習	ランダム (各単語 5 点)	ランダム (全体から)	全対象単語に対する	領域間距離を利用	負の転移を考慮
OC → PB	78.89	77.85	78.33	78.45	77.36	78.86
PB → PN	83.37	82.47	78.33	84.62	83.13	83.95
PN → OC	74.82	73.94	74.85	74.99	72.88	75.66
OC → PN	75.60	74.80	77.49	67.00	65.26	76.18
PN → PB	82.09	81.02	74.69	81.65	83.03	81.99
PB → OC	78.70	76.35	74.69	78.20	74.57	78.95

第 8 章　結論

本論文では WSD の領域適応の問題に対して能動学習を試みた．WSD を対象とした能動学習では、各対象単語毎に能動学習を適用する形になっているが，この形では WSD の総合的な評価であるマクロ平均やマイクロ平均の評価値を効率的に向上させることができる保証はない．

そこで，WSD の全対象単語の全テストデータを対象にしてラベル付けするデータを選択する手法と，領域間距離を考慮した信頼度を用いる手法を試した．BCCWJ コーパスの OC(Yahoo!知恵袋)，PB(書籍) 及び PN(新聞) の 3 つの領域を用いた実験の結果，全対象単語から選択する手法は効果があったが，領域間距離を考慮した手法には効果は見られなかった．

そこで，仮に領域間距離を正しく測れているとして負の転移の有無を判定し，その判定を用いてデータ選択の手順を変える手法を試した結果，領域間距離を利用するという手法には効果が有ることが確認できた．

提案手法を効果的に利用するには，領域間距離を正しく測る方法と算出した領域間距離をラベル付けのデータ選択に取り入れる方法を考える必要がある．これらが今後の課題である．

謝辞

本研究を進めるにあたり、多くのご指導ご協力を頂いた指導教員の新納浩幸准教授に感謝致します。また、日常の議論を通じて多くの知識や示唆を頂いた新納研究室の皆様に感謝します。

参考文献

[1] Olivier Chapelle, Bernhard Sch ¨ olkopf, Alexander Zien et al. (2006) Semi-supervised learning, Vol. 2: MIT press Cambridge.

[2] Kikuo Maekawa (2007) “Design of a Balanced Corpus of Contemporary Written Japanese,” in Symposium on Large-Scale Knowledge Resources (LKR2007), pp. 55–58.

[3] Manabu Okumura, Kiyoaki Shirai, Kanako Komiya, and Hikaru Yokono (2010) “SemEval-2010 Task: Japanese WSD,” in The 5th International Workshop on Semantic Evaluation, pp. 69–74.

[4] Burr Settles (2010) “Active learning literature survey,” University of Wisconsin, Madison.

[5] Anders Sogaard (2013) Semi-Supervised Learning and Domain Adaptation in Natural Language Processing: Morgan Claypool.

[6] 神嶌敏弘 (2010)「転移学習」，人工知能学会誌，第 25 巻，第 4 号，pp.572–580.

[7] 小野寺喜行, 新納浩幸 (2013) “クラスタリングを利用した能動学習による語義曖昧性解消の領域適応,” 第 4 回日本語学ワークショップ,pp.309-316.

[8] 小野寺喜行, 新納浩幸 (2014) “領域間距離を利用した能動学習による語義曖昧性解消の領域適応,” 第 5 回日本語学ワークショップ,to appear.

[9] 中谷秀洋 (2013) “Active Learning 入門, 2013/9/1 DSIRNLP # 4, 中谷 秀洋@サイボウズ・ラボ / @shuyo,” http://www.slideshare.net/shuyo/introduction-to-active-learning-25787487

付録 A ソースリスト

```
1  #!/usr/bin/perl
2
3  use strict;
4  use warnings;
5  use utf8;
6  use Encode qw/decode encode/;
7  binmode STDOUT, ":utf8";
8  my @xx=(1707,2998,7479,10487,17877,20676,22038,35478,37713,
9    40333,40699,48488,50038,51332,52310,52744);
10 #各ディレクトリ下のファイルlabelを読込してlabel_testとrabel_teachのsvm形式を作る
11 foreach my $name (@xx){
12
13 #読込ファイルを開く
14     my $filenameoc = "$name/OC/label";
15     open my $fhoc, '<', $filenameoc
16   or die qq/Can't open file "$filenameoc" : $!/;
17
18     my $filenamepb = "$name/PB/label";
19     open my $fhpb, '<', $filenamepb
20   or die qq/Can't open file "$filenamepb" : $!/;
21
22     my $filenamepn = "$name/PN/label";
23     open my $fhpn, '<', $filenamepn
24   or die qq/Can't open file "$filenamepn" : $!/;
25
26 #書込みファイルを開く
27     my $wfile ="$name/words_list";
28     open my $wfh, '>', $wfile
29   or die qq/Can't open file "$wfile" : $!/;
30
31 #label->まとめて処理
32     read $fhoc,my $oc,(-s "$filenameoc");
33     $oc=decode('UTF-8',$oc);
34
35     read $fhpb,my $pb,(-s "$filenamepb");
36     $pb=decode('UTF-8',$pb);
37
38     read $fhpn,my $pn,(-s "$filenamepn");
39     $pn=decode('UTF-8',$pn);
40
```

```
41 #label を抜出
42     my @aa=$oc=~m/([^ ]+)\n/g;
43     my @bb=$pb=~m/([^ ]+)\n/g;
44     my @cc=$pn=~m/([^ ]+)\n/g;
45     my %dd=();
46
47     foreach my $w (@aa){
48   $dd{$w}++;
49     }
50
51     foreach my $w (@bb){
52   $dd{$w}++;
53     }
54
55     foreach my $w (@cc){
56   $dd{$w}++;
57     }
58
59     my $wfile2 ="$name/svm_label";
60     open my $wfh2, '>', $wfile2
61   or die qq/Can't open file "$wfile2" : $!/;
62
63     my $pp=1;
64     foreach my $gg(keys %dd){
65   print $wfh2 "$pp $gg\n";
66   $pp++;
67     }
68
69     close $wfh2 or die qw/Can't close file "$wfile2": $!/;
70
71 #単語を抜出
72     my @tmp_oc=$oc=~m/(?:e\d=(.+?)\s)/g;
73     my @tmp_pb=$pb=~m/(?:e\d=(.+?)\s)/g;
74     my @tmp_pn=$pn=~m/(?:e\d=(.+?)\s)/g;
75
76 #単語->hash{単語}=頻度
77     my %hash=();
78
79     foreach my $word (@tmp_oc){
80   $hash{$word}++;
81     }
82
83     foreach my $word (@tmp_pb){
84   $hash{$word}++;
85     }
86
87     foreach my $word (@tmp_pn){
88   $hash{$word}++;
89     }
90
```

```
91 #書込み
92     my $num=1;
93
94     foreach my $yy (keys %hash){
95   print $wfh "$num ".encode('UTF-8',$yy)."\n";
96   $num++;
97     }
98
99     close $fhoc;
100    close $fhpb;
101    close $fhpn;
102    close $wfh or die qw/Can't close file "$wfile": $!/;
103 }
```

ソースコード 1　コーパスに出現する単語に INDEX 番号を付けたファイルを作成するプログラム

```
1  #!/usr/bin/perl
2
3  use strict;
4  use warnings;
5  use utf8;
6  use Encode qw/decode encode/;
7  binmode STDOUT, ":utf8";
8  my @xx=(1707,2998,7479,10487,17877,20676,22038,35478,37713,
9    40333,40699,48488,50038,51332,52310,52744);
10 #各ディレクトリ下のファイルlabelを読込してlabel_testとrabel_teachのsvm形式を作る
11 foreach my $name (@xx){
12 #OCの処理
13     my $g='OC';
14     make_label($name,$g);
15 #PBの処理
16     $g='PB';
17     make_label($name,$g);
18 #PNの処理
19 #     $g='PN';
20 #     make_label($name,$g);
21 }
22
23 sub make_label{
24
25     my($name,$g)=@_;
26 #読込ファイルを開く
27     my $filename ="$name/$g/label";
28     open my $fh, '<', $filename
29         or die qq/Can't open file "$filename" : $!/;
30     my $filename2 ="$name/words_list";
31     open my $fh2, '<', $filename2
32         or die qq/Can't open file "$filename2" : $!/;
33     my $filename3 ="$name/svm_label";
34     open my $fh3, '<', $filename3
```

```
35          or die qq/Can't open file "$filename3" : $!/;
36
37  #INDEX->hash_index
38      my %hash_index=();
39      while(my $line=<$fh2>){
40    $line=decode('UTF-8',$line);
41    $line=~m/(\d+)\s(.+?)\n/;
42    $hash_index{$2}=$1;
43      }
44
45  #label->hash_label
46      my %hash_label=();
47      while(my $line=<$fh3>){
48    $line=decode('UTF-8',$line);
49    $line=~m/(\d+)\s(.+?)\n/;
50    $hash_label{$2}=$1;
51      }
52
53  #書込みファイルを開く
54      my $wfile ="$name/label_test_$g";
55      open my $wfh, '>', $wfile
56    or die qq/Can't open file "$wfile" : $!/;
57      my $wfile2 ="$name/label_teach_$g";
58      open my $wfh2, '>', $wfile2
59    or die qq/Can't open file "$wfile2" : $!/;
60
61  #「label」->1行ずつ処理
62      while(my $line=<$fh>){
63    $line=decode('UTF-8',$line);
64  #単語を抜出
65    my @tmp=$line=~m/(?:e\d=(.+?)\s)/g;
66  #labelを抜出
67    my @temp=$line=~m/([^ ]+)\n/;
68    my $label=$hash_label{$temp[0]};
69  #単語->hash{index}=頻度
70    my %hash=();
71    foreach my $word (@tmp){
72        $hash{$hash_index{$word}}++;
73       # print "word->$word index->$hash_index{$word}";
74    }
75    #print "$#tmp label->$label\n";
76  #書込み
77    print $wfh encode('UTF-8',$label);
78    print $wfh2 encode('UTF-8',$label);
79    foreach my $num (sort { $a <=> $b } keys %hash){
80        print $wfh " ".encode('UTF-8',$num).":$hash{$num}";
81        print $wfh2 " ".encode('UTF-8',$num).":$hash{$num}";
82    }
83    print $wfh "\n";
84    print $wfh2 "\n";
```

```
85      }
86      close $fh;
87      close $fh2;
88      close $wfh or die qw/Can't close file "$wfile": $!/;
89      close $wfh2 or die qw/Can't close file "$wfile2": $!/;
90
91
92  }
```

ソースコード 2　コーパスから SVM 用のファイルを作成するプログラム

```
1   #!/usr/bin/perl
2
3   use strict;
4   use warnings;
5
6   use utf8;
7   use Encode qw/decode encode/;
8   use List::Util 'max';
9   binmode STDOUT, ":utf8";
10
11  my $num = shift;
12  my $numminus = $num - 1;
13
14  my @words=(1707,2998,7479,10487,17877,20676,22038,35478,
15        37713,40333,40699,48488,50038,51332,52310,52744);
16
17  foreach my $word (@words){
18
19      my $g = 'OC';
20      my $h = 'PB';
21      svm_plus($word,$g,$h,$numminus);
22
23      $g = 'PB';
24      $h = 'OC';
25      svm_plus($word,$g,$h,$numminus);
26
27  #     my $g = 'OC';
28  #     my $h = 'PN';
29  #     svm_plus($word,$g,$h,$numminus);
30
31  #     $g = 'PN';
32  #     $h = 'OC';
33  #     svm_plus($word,$g,$h,$numminus);
34
35  #     my $g = 'PB';
36  #     my $h = 'PN';
37  #     svm_plus($word,$g,$h,$numminus);
38
39  #     $g = 'PN';
```

```
40 #     $h = 'PB';
41 #     svm_plus($word,$g,$h,$numminus);
42
43 }
44
45 sub svm_plus{
46
47     my($word,$g,$h,$n) = @_; #単語,訓練,学習
48
49     #テストデータのラベル
50     my $fts = "${word}/label_test_${h}";
51     open my $fhts, "<", $fts
52   or die "Cannot open $fts: $!";
53
54     #テスト結果の信頼度
55     my $frs = "${word}/result${g}${h}${word}${n}";
56     open my $fhrs, "<", $frs
57   or die "Cannot open $frs: $!";
58
59     #ヘッダを取る
60     my $rsh = <$fhrs>;
61
62     my %lines;
63
64     while(my $tline = <$fhts>){    #テストデータから1行読み取り
65
66   chomp $tline;
67   $tline = decode('UTF-8', $tline);
68
69   my $rline = <$fhrs>;       #信頼度のデータから1行読み取り
70
71   #chomp $rline;
72   $rline = decode('UTF-8', $rline);
73
74   my @rates;
75
76   while( $rline =~ /\s((?:\d\.\d+|\d))/g ){     #読み込んだ行の信頼度を全て取り出す
77       if( $1 >= 0 && $1 <= 1 ){
78     push @rates, $1;
79       }else{
80     push @rates, 0;
81       }
82   }
83
84   $lines{max @rates} = $tline;     #最も高いものが学習結果の信頼度
85                                    #その値をキーとしてハッシュに格納
86
87     }
88
89     close $fhts;
```

```
90      close $fhrs;
91
92      my @lines2;
93
94      #信頼度が低い順に並べ替え
95      foreach my $key ( sort { $a <=> $b } keys %lines ){
96    push(@lines2, $lines{$key});
97      }
98
99      #信頼度が最も低いデータを取り出す
100     my $s = shift @lines2;
101
102     #訓練データのラベル
103     my $ftc = "${word}/label_teach_${g}";
104     open my $fhtc, "<", $ftc
105   or die "Cannot open $ftc: $!";
106
107     my @svm;
108
109     while( my $line = <$fhtc> ){    #1行ずつ読み込み配列に格納
110
111   chomp $line;
112   $line = decode('UTF-8', $line);
113
114   push(@svm, $line);
115
116     }
117
118     close $fhtc;
119
120     #出力用ファイル
121     my $ftcp = "${word}/label_teach_${g}";
122     open my $fhtcp, ">", $ftcp
123   or die "Cannot open $ftcp: $!";
124
125     #読み込んだ訓練データを出力
126     foreach my $data (@svm){
127
128   $data = encode('UTF-8', $data);
129   print $fhtcp $data,"\n";
130
131     }
132
133     #信頼度が最も低いデータを出力
134     $s = encode('UTF-8', $s);
135     print $fhtcp $s,"\n";
136
137 }
```

ソースコード 3　各単語で信頼度最低のデータを訓練データに追加するプログラム（OC と PB の領域適応の場合）

```
#!/usr/bin/perl

use strict;
use warnings;

use utf8;
use Encode qw/decode encode/;
use List::Util 'max';
binmode STDOUT, ":utf8";

my $num = shift;
my $numminus = $num - 1;

print "${numminus}\n";

my @words=(1707,2998,7479,10487,17877,20676,22038,35478,
     37713,40333,40699,48488,50038,51332,52310,52744);

#  信頼度と単語  単語とラベル
my %rwocpb; my %wlocpb;
my %rwpboc; my %wlpboc;
#my %rwocpn; my %wlocpn;
#my %rwpnoc; my %wlpnoc;
#my %rwpbpn; my %wlpbpn;
#my %rwpnpb; my %wlpnpb;

foreach my $word (@words){

    my $g = 'OC';
    my $h = 'PB';
    my $r = low_label($word,$g,$h,$numminus);
    my @r = @$r;
    $rwocpb{$r[1]} = $word;
    $wlocpb{$word} = $r[0];

    $g = 'PB';
    $h = 'OC';
    $r = low_label($word,$g,$h,$numminus);
    @r = @$r;
    $rwpboc{$r[1]} = $word;
    $wlpboc{$word} = $r[0];

#    my $g = 'OC';
#    my $h = 'PN';
#    my $r = low_label($word,$g,$h,$num_minus);
#    my @r = @$r;
#    $rwocpn{$r[1]} = $word;
#    $wlocpn{$word} = $r[0];

```

```
50 #     $g = 'PN';
51 #     $h = 'OC';
52 #     $r = low_label($word,$g,$h,$num_minus);
53 #     @r = @$r;
54 #     $rwpnoc{$r[1]} = $word;
55 #     $wlpnoc{$word} = $r[0];
56 
57 #     my $g = 'PB';
58 #     my $h = 'PN';
59 #     my $r = low_label($word,$g,$h,$num_minus);
60 #     my @r = @$r;
61 #     $rwpbpn{$r[1]} = $word;
62 #     $wlpbpn{$word} = $r[0];
63 
64 #     $g = 'PN';
65 #     $h = 'PB';
66 #     $r = low_label($word,$g,$h,$num_minus);
67 #     @r = @$r;
68 #     $rwpnpb{$r[1]} = $word;
69 #     $wlpnpb{$word} = $r[0];
70 
71 }
72 
73 #追加したラベルのログ用ファイル
74 my $flog = "log/plus${num}";
75 open my $fhlog, ">", $flog
76     or die "Cannot open $flog: $!";
77 
78 #信頼度が低い順に配列に格納
79 
80 my @wocpb;  #単語
81 my @locpb;  #ラベル
82 
83 foreach my $key ( sort { $a <=> $b } keys %rwocpb ){
84     my $w = $rwocpb{$key};
85     push(@wocpb, $w);
86     push(@locpb, $wlocpb{$w});
87 }
88 
89 my $lw = $wocpb[0];  #最も信頼度が低いラベルとその単語
90 my $ll = $locpb[0];
91 my $g = 'OC';
92 my $h = 'PB';
93 label_plus($lw,$g,$h,$ll,$fhlog);
94 
95 my @wpboc;  #単語
96 my @lpboc;  #ラベル
97 
98 foreach my $key ( sort { $a <=> $b } keys %rwpboc ){
99     my $w = $rwpboc{$key};
```

```
100     push(@wpboc, $w);
101     push(@lpboc, $wlpboc{$w});
102 }
103 
104 $lw = $wpboc[0];  #最も信頼度が低いラベルとその単語
105 $ll = $lpboc[0];
106 $g = 'PB';
107 $h = 'OC';
108 label_plus($lw,$g,$h,$ll,$fhlog);
109 
110 #my @wocpn;  #単語
111 #my @locpn;  #ラベル
112 
113 #foreach my $key ( sort { $a <=> $b } keys %rwocpn ){
114 #     my $w = $rwocpn{$key};
115 #     push(@wocpn, $w);
116 #     push(@locpn, $wlocpn{$w});
117 #}
118 
119 #my $lw = $wocpn[0];  #最も信頼度が低いラベルとその単語
120 #my $ll = $locpn[0];
121 #my $g = 'OC';
122 #my $h = 'PN';
123 #label_plus($lw,$g,$h,$ll,$fhlog);
124 
125 #my @wpnoc;  #単語
126 #my @lpnoc;  #ラベル
127 
128 #foreach my $key ( sort { $a <=> $b } keys %rwpnoc ){
129 #     my $w = $rwpnoc{$key};
130 #     push(@wpnoc, $w);
131 #     push(@lpnoc, $wlpnoc{$w});
132 #}
133 
134 #$lw = $wpnoc[0];  #最も信頼度が低いラベルとその単語
135 #$ll = $lpnoc[0];
136 #$g = 'PN';
137 #$h = 'OC';
138 #label_plus($lw,$g,$h,$ll,$fhlog);
139 
140 #my @wpbpn;  #単語
141 #my @lpbpn;  #ラベル
142 
143 #foreach my $key ( sort { $a <=> $b } keys %rwpbpn ){
144 #     my $w = $rwpbpn{$key};
145 #     push(@wpbpn, $w);
146 #     push(@lpbpn, $wlpbpn{$w});
147 #}
148 
149 #my $lw = $wpbpn[0];  #最も信頼度が低いラベルとその単語
```

```
150 #my $ll = $lpbpn[0];
151 #my $g = 'PB';
152 #my $h = 'PN';
153 #label_plus($lw,$g,$h,$ll,$fhlog);
154
155 #my @wpnpb;  #単語
156 #my @lpnpb;  #ラベル
157
158 #foreach my $key ( sort { $a <=> $b } keys %rwpnpb ){
159 #    my $w = $rwpnpb{$key};
160 #    push(@wpnpb, $w);
161 #    push(@lpnpb, $wlpnpb{$w});
162 #}
163
164 #$lw = $wpnpb[0];  #最も信頼度が低いラベルとその単語
165 #$ll = $lpnpb[0];
166 #$g = 'PN';
167 #$h = 'PB';
168 #label_plus($lw,$g,$h,$ll,$fhlog);
169
170 close $fhlog;
171
172 sub low_label{
173
174     my($word,$g,$h,$n) = @_; #単語,訓練,学習
175
176     #テストデータのラベル
177     my $fts = "${word}/label_test_${h}";
178     open my $fhts, "<", $fts
179   or die "Cannot open $fts: $!";
180
181     #テスト結果の信頼度
182     my $frs = "${word}/result${g}${h}${word}${n}";
183     open my $fhrs, "<", $frs
184   or die "Cannot open $frs: $!";
185
186     #ヘッダを取る
187     my $rsh = <$fhrs>;
188
189     my %lines;
190
191     while(my $tline = <$fhts>){    #テストデータから1行読み取り
192
193   chomp $tline;
194   $tline = decode('UTF-8', $tline);
195
196   my $rline = <$fhrs>;       #信頼度のデータから1行読み取り
197
198   $rline = decode('UTF-8', $rline);
199
```

```
200     my @rates;
201
202     while( $rline =~ /\s((?:\d\.\d+|\d))/g ){     #読み込んだ行の信頼度を全て取り出す
203         if( $1 >= 0 && $1 <= 1 ){
204       push @rates, $1;
205         }else{
206       push @rates, 0;
207         }
208     }
209
210     $lines{max @rates} = $tline;     #最も高いものが学習結果の信頼度
211                                      #その値をキーとしてハッシュに格納
212
213       }
214
215       close $fhts;
216       close $fhrs;
217
218       my @lines2;
219       my @lines3;
220
221       #信頼度が低い順に並べ替え
222       foreach my $key ( sort { $a <=> $b } keys %lines ){
223     push(@lines2, $lines{$key});    #ラベル
224     push(@lines3, $key);            #信頼度
225       }
226
227       #信頼度が最も低いデータを取り出す
228       my @r;
229       $r[0] = shift @lines2;
230       $r[1] = shift @lines3;
231
232       my $r = \@r;
233       return $r;
234
235 }
236
237 sub label_plus{
238
239       my($word, $g, $h, $s, $fhlog) = @_;
240
241       #訓練データのラベル
242       my $ftc = "${word}/label_teach_${g}";
243       open my $fhtc, "<", $ftc
244     or die "Cannot open $ftc: $!";
245
246       my @svm;
247
248       while( my $line = <$fhtc> ){    #1行ずつ読み込み配列に格納
249
```

```
250   chomp $line;
251   $line = decode('UTF-8', $line);
252
253   push(@svm, $line);
254
255     }
256
257     close $fhtc;
258
259     #出力用ファイル
260     my $ftcp = "${word}/label_teach_${g}";
261     open my $fhtcp, ">", $ftcp
262   or die "Cannot open $ftcp: $!";
263
264     #読み込んだ訓練データを出力
265     foreach my $data (@svm){
266
267   $data = encode('UTF-8', $data);
268   print $fhtcp $data,"\n";
269
270     }
271
272     #信頼度が最も低いデータを出力
273     $s = encode('UTF-8', $s);
274     print $fhtcp $s,"\n";
275
276     #ログ用ファイルに出力"
277     my $outlog = "${g}${h} : ${word} : ${s}";
278     $outlog = encode('UTF-8', $outlog);
279     print $fhlog $outlog,"\n";
280
281 }
```

ソースコード 4　対象単語全体で信頼度最低のデータを訓練データに追加するプログラム（OC と PB の領域適応の場合）

```
 1 #!/usr/bin/perl
 2
 3 use strict;
 4 use warnings;
 5
 6 use utf8;
 7 use Encode qw/decode encode/;
 8 use List::Util 'max';
 9 use List::Util 'sum';
10 binmode STDOUT, ":utf8";
11
12 my $num = shift;
13 my $numminus = $num - 1;
14
```

```
15 print "${numminus}\n";
16
17 my @words=(1707,2998,7479,10487,17877,20676,22038,35478,
18      37713,40333,40699,48488,50038,51332,52310,52744);
19
20 #  信頼度と単語  単語とラベル
21 my %rwocpb; my %wlocpb;
22 my %rwpboc; my %wlpboc;
23 #my %rwocpn; my %wlocpn;
24 #my %rwpnoc; my %wlpnoc;
25 #my %rwpbpn; my %wlpbpn;
26 #my %rwpnpb; my %wlpnpb;
27
28 foreach my $word (@words){
29
30     my $g = 'OC';
31     my $h = 'PB';
32     my $r = low_label($word,$g,$h,$numminus);
33     my @r = @$r;
34     $rwocpb{$r[1]} = $word;
35     $wlocpb{$word} = $r[0];
36
37     $g = 'PB';
38     $h = 'OC';
39     $r = low_label($word,$g,$h,$numminus);
40     @r = @$r;
41     $rwpboc{$r[1]} = $word;
42     $wlpboc{$word} = $r[0];
43
44 #     $g = 'OC';
45 #     $h = 'PN';
46 #     $r = low_label($word,$g,$h,$num_minus);
47 #     @r = @$r;
48 #     $rwocpn{$r[1]} = $word;
49 #     $wlocpn{$word} = $r[0];
50
51 #     $g = 'PN';
52 #     $h = 'OC';
53 #     $r = low_label($word,$g,$h,$num_minus);
54 #     @r = @$r;
55 #     $rwpnoc{$r[1]} = $word;
56 #     $wlpnoc{$word} = $r[0];
57
58 #     $g = 'PB';
59 #     $h = 'PN';
60 #     $r = low_label($word,$g,$h,$num_minus);
61 #     @r = @$r;
62 #     $rwpbpn{$r[1]} = $word;
63 #     $wlpbpn{$word} = $r[0];
64
```

```
65 #     $g = 'PN';
66 #     $h = 'PB';
67 #     $r = low_label($word,$g,$h,$num_minus);
68 #     @r = @$r;
69 #     $rwpnpb{$r[1]} = $word;
70 #     $wlpnpb{$word} = $r[0];
71
72 }
73
74 #追加したラベルのログ用ファイル
75 my $flog = "log/plus${num}";
76 open my $fhlog, ">", $flog
77     or die "Cannot open $flog: $!";
78
79 #評価値が低い順に配列に格納
80
81 my @wocpb;  #単語
82 my @locpb;  #ラベル
83
84 foreach my $key ( sort { $a <=> $b } keys %rwocpb ){
85     my $w = $rwocpb{$key};
86     push(@wocpb, $w);
87     push(@locpb, $wlocpb{$w});
88 }
89
90 my $lw = $wocpb[0];  #最も評価値が低いラベルとその単語
91 my $ll = $locpb[0];
92 my $g = 'OC';
93 my $h = 'PB';
94 label_plus($lw,$g,$h,$ll,$fhlog);
95
96 my @wpboc;  #単語
97 my @lpboc;  #ラベル
98
99 foreach my $key ( sort { $a <=> $b } keys %rwpboc ){
100     my $w = $rwpboc{$key};
101     push(@wpboc, $w);
102     push(@lpboc, $wlpboc{$w});
103 }
104
105 $lw = $wpboc[0];
106 $ll = $lpboc[0];
107 $g = 'PB';
108 $h = 'OC';
109 label_plus($lw,$g,$h,$ll,$fhlog);
110
111 #my @wocpn;  #単語
112 #my @locpn;  #ラベル
113
114 #foreach my $key ( sort { $a <=> $b } keys %rwocpn ){
```

```
115 #     my $w = $rwocpn{$key};
116 #     push(@wocpn, $w);
117 #     push(@locpn, $wlocpn{$w});
118 #}
119
120 #my $lw = $wocpn[0];
121 #my $ll = $locpn[0];
122 #$g = 'OC';
123 #$h = 'PN';
124 #label_plus($lw,$g,$h,$ll,$fhlog);
125
126 #my @wpnoc;  #単語
127 #my @lpnoc;  #ラベル
128
129 #foreach my $key ( sort { $a <=> $b } keys %rwpnoc ){
130 #     my $w = $rwpnoc{$key};
131 #     push(@wpnoc, $w);
132 #     push(@lpnoc, $wlpnoc{$w});
133 #}
134
135 #$lw = $wpnoc[0];
136 #$ll = $lpnoc[0];
137 #$g = 'PN';
138 #$h = 'OC';
139 #label_plus($lw,$g,$h,$ll,$fhlog);
140
141 #my @wpbpn;  #単語
142 #my @lpbpn;  #ラベル
143
144 #foreach my $key ( sort { $a <=> $b } keys %rwpbpn ){
145 #     my $w = $rwpbpn{$key};
146 #     push(@wpbpn, $w);
147 #     push(@lpbpn, $wlpbpn{$w});
148 #}
149
150 #$lw = $wpbpn[0];
151 #$ll = $lpbpn[0];
152 #$g = 'PB';
153 #$h = 'PN';
154 #label_plus($lw,$g,$h,$ll,$fhlog);
155
156 #my @wpnpb;  #単語
157 #my @lpnpb;  #ラベル
158
159 #foreach my $key ( sort { $a <=> $b } keys %rwpnpb ){
160 #     my $w = $rwpnpb{$key};
161 #     push(@wpnpb, $w);
162 #     push(@lpnpb, $wlpnpb{$w});
163 #}
164
```

```
#$lw = $wpnpb[0];
#$ll = $lpnpb[0];
#$g = 'PN';
#$h = 'PB';
#label_plus($lw,$g,$h,$ll,$fhlog);

close $fhlog;

sub low_label{

    my($word,$g,$h,$n) = @_; #単語,訓練,学習

    #テストデータのラベル
    my $fts = "${word}/label_test_${h}";
    open my $fhts, "<", $fts
  or die "Cannot open $fts: $!";

    #テスト結果の信頼度一覧
    my $frs = "${word}/result${g}${h}${word}${n}";
    open my $fhrs, "<", $frs
  or die "Cannot open $frs: $!";

    #テストデータと訓練データの類似度と組成一覧
    my $fdr = "${word}/${h}-${g}-dr";
    open my $fhdr, "<", $fdr
  or die "Cannot open $fdr: $!";

    #ヘッダを取る
    my $rsh = <$fhrs>;

    #ラベルを配列に格納
    my @linets;
    while(my $line = <$fhts>){
  chomp $line;
  $line = decode('UTF-8', $line);
  push @linets, $line;
    }

    #信頼度一覧を配列に格納
    my @liners;
    while(my $line = <$fhrs>){
  chomp $line;
  $line = decode('UTF-8', $line);
  push @liners, $line;
    }

    #類似度と組成一覧を配列に格納
    my @linedr;
    while(my $line = <$fhdr>){
  chomp $line;
```

```
215   $line = decode('UTF-8', $line);
216   push @linedr, $line;
217     }
218
219     #"m"の計算
220     my @hj;
221     foreach my $line (@liners){
222   my @rates;
223   while( $line =~ /\s((?:\d\.\d+|\d))/g){
224       if( $1 >= 0 && $1 <= 1 ){
225     push @rates, $1;
226       }else{
227     push @rates, 0;
228       }
229   }
230   push @hj, max @rates;
231     }
232     my $num = @hj;   #要素数
233     my $m = (sum @hj) / $num;
234
235     #"u^2"の計算
236     my $u2 = 0;
237     foreach my $h (@hj){
238   $u2 += ( ($m - $h) * ($m - $h) );
239     }
240     $u2 /= ($num - 1);
241
242     #"m'"の計算
243     my @sj;
244     foreach my $line(@linedr){
245   if($line =~ /(\d+\.\d+)\s/){
246       push @sj, $1;
247   }
248     }
249     my $md = (sum @sj) / $num;
250
251     #"u'^2"の計算
252     my $ud2 = 0;
253     foreach my $s (@sj){
254   $ud2 += ( ($md - $s) * ($md - $s) );
255     }
256     $ud2 /= ($num - 1);
257
258     #r_j,w_j,x_jの計算
259     my %xj;
260     my $u = sqrt $u2;
261     my $ud = sqrt $ud2;
262     for( my $i = 0; $i < $num; $i++ ){
263   my $r = ( $hj[$i] - $m ) / $u;
264   my $w = ( $u * ( $sj[$i] - $md ) ) / $ud + $m;
```

```
265     my $x = $r + $w;
266     $xj{$x} = $linets[$i];
267       }
268
269       close $fhts;
270       close $fhrs;
271       close $fhdr;
272
273       my @xj2;
274       my @xj3;
275
276       #評価値が低い順に並べ替え
277       foreach my $key ( sort { $a <=> $b } keys %xj ){
278     push(@xj2, $xj{$key});    #ラベル
279     push(@xj3, $key);             #信頼度
280       }
281
282       #評価値が最も低いデータを取り出す
283       my @r;
284       $r[0] = shift @xj2;
285       $r[1] = shift @xj3;
286
287       my $r = \@r;
288       return $r;
289
290 }
291
292 sub label_plus{
293
294       my($word, $g, $h, $s, $fhlog) = @_;
295
296       #訓練データのラベル
297       my $ftc = "${word}/label_teach_${g}";
298       open my $fhtc, "<", $ftc
299     or die "Cannot open $ftc: $!";
300
301       my @svm;
302
303       while( my $line = <$fhtc> ){    #1行ずつ読み込み配列に格納
304
305     chomp $line;
306     $line = decode('UTF-8', $line);
307
308     push(@svm, $line);
309
310       }
311
312       close $fhtc;
313
314       #出力用ファイル
```

```
315     my $ftcp = "${word}/label_teach_${g}";
316     open my $fhtcp, ">", $ftcp
317   or die "Cannot open $ftcp: $!";
318
319     #読み込んだ訓練データを出力
320     foreach my $data (@svm){
321
322   $data = encode('UTF-8', $data);
323   print $fhtcp $data,"\n";
324
325     }
326
327     #信頼度が最も低いデータを出力
328     $s = encode('UTF-8', $s);
329     print $fhtcp $s,"\n";
330
331     #ログ用ファイルに出力"
332     my $outlog = "${g}${h} : ${word} : ${s}";
333     $outlog = encode('UTF-8', $outlog);
334     print $fhlog $outlog,"\n";
335
336 }
```

ソースコード 5　対象単語全体で評価値最低のデータを訓練データに追加するプログラム（OC と PB の領域適応の場合）

```
1  #!/usr/bin/perl
2
3  use strict;
4  use warnings;
5
6  use utf8;
7  use Encode qw/decode encode/;
8  use List::Util 'max';
9  binmode STDOUT, ":utf8";
10
11 my $num = shift;
12 my $numminus = $num - 1;
13
14 my $g;
15 my $h;
16
17 my @wocpb=(7479,10487,20676,22038,50038,51332);
18 foreach my $word (@wocpb){
19     $g = 'OC';
20     $h = 'PB';
21     svm_plus($word,$g,$h,$numminus);
22 }
23
24 #my @wpboc=(2998,35478,37713,40333,40699,51332,52310);
```

```
25 #foreach my $word (@wpboc){
26 #    $g = 'PB';
27 #    $h = 'OC';
28 #    svm_plus($word,$g,$h,$numminus);
29 #}
30
31 #my @wocpn=(1707,2998,7479,35478,52310,52744);
32 #foreach my $word (@wocpn){
33 #    $g = 'OC';
34 #    $h = 'PN';
35 #    svm_plus($word,$g,$h,$numminus);
36 #}
37
38 #my @wpnoc=(1707,2998,17877,20676,37713,40333,40699);
39 #foreach my $word (@wpnoc){
40 #    $g = 'PN';
41 #    $h = 'OC';
42 #    svm_plus($word,$g,$h,$numminus);
43 #}
44
45 #my @wpbpn=(1707,2998,22038,40333,40699,48488,51332,52310,52744);
46 #foreach my $word (@wpbpn){
47 #    $g = 'PB';
48 #    $h = 'PN';
49 #    svm_plus($word,$g,$h,$numminus);
50 #}
51
52 #my @wpnpb=(1707,2998,7479,17877,20676,37713,40333,40699,52744);
53 #foreach my $word (@wpnpb){
54 #    $g = 'PN';
55 #    $h = 'PB';
56 #    svm_plus($word,$g,$h,$numminus);
57 #}
58
59
60 sub svm_plus{
61
62     my($word,$g,$h,$n) = @_; #単語,訓練,学習
63
64     #テストデータのラベル
65     my $fts = "${word}/label_test_${h}";
66     open my $fhts, "<", $fts
67   or die "Cannot open $fts: $!";
68
69     #テスト結果の信頼度
70     my $frs = "${word}/result${g}${h}${word}${n}";
71     open my $fhrs, "<", $frs
72   or die "Cannot open $frs: $!";
73
74     #ヘッダを取る
```

```
75      my $rsh = <$fhrs>;
76
77      my %lines;
78
79      while(my $tline = <$fhts>){    #テストデータから1行読み取り
80
81    chomp $tline;
82    $tline = decode('UTF-8', $tline);
83
84    my $rline = <$fhrs>;       #信頼度のデータから1行読み取り
85
86    #chomp $rline;
87    $rline = decode('UTF-8', $rline);
88
89    my @rates;
90
91    while( $rline =~ /\s((?:\d\.\d+|\d))/g ){     #読み込んだ行の信頼度を全て取り出す
92        if( $1 >= 0 && $1 <= 1 ){
93      push @rates, $1;
94        }else{
95      push @rates, 0;
96        }
97    }
98
99    $lines{max @rates} = $tline;     #最も高いものが学習結果の信頼度
100                                     #その値をキーとしてハッシュに格納
101
102     }
103
104     close $fhts;
105     close $fhrs;
106
107     my @lines2;
108
109     #信頼度が低い順に並べ替え
110     foreach my $key ( sort { $a <=> $b } keys %lines ){
111   push(@lines2, $lines{$key});
112     }
113
114     #信頼度が最も低いデータを取り出す
115     my $s = shift @lines2;
116
117     #訓練データのラベル
118     my $ftc = "${word}/label_teach_${g}";
119     open my $fhtc, "<", $ftc
120   or die "Cannot open $ftc: $!";
121
122     my @svm;
123
124     while( my $line = <$fhtc> ){    #1行ずつ読み込み配列に格納
```

```
125
126   chomp $line;
127   $line = decode('UTF-8', $line);
128
129   push(@svm, $line);
130
131     }
132
133     close $fhtc;
134
135     #出力用ファイル
136     my $ftcp = "${word}/label_teach_${g}";
137     open my $fhtcp, ">", $ftcp
138   or die "Cannot open $ftcp: $!";
139
140     #読み込んだ訓練データを出力
141     foreach my $data (@svm){
142
143   $data = encode('UTF-8', $data);
144   print $fhtcp $data,"\n";
145
146     }
147
148     #信頼度が最も低いデータを出力
149     $s = encode('UTF-8', $s);
150     print $fhtcp $s,"\n";
151
152 }
```

ソースコード 6　負の転移無しの各単語で信頼度最低のデータを訓練データに追加するプログラム（OC と PB の領域適応の場合）

```
1  #!/usr/bin/perl
2
3  use strict;
4  use warnings;
5
6  use utf8;
7  use Encode qw/decode encode/;
8  use List::Util 'max';
9  binmode STDOUT, ":utf8";
10
11 my $num = shift;
12 my $numminus = $num - 1;
13
14 #  信頼度と単語  単語とラベル
15 my %rwocpb; my %wlocpb;
16 #my %rwpboc; my %wlpboc;
17 #my %rwocpn; my %wlocpn;
18 #my %rwpnoc; my %wlpnoc;
```

```
#my %rwpbpn; my %wlpbpn;
#my %rwpnpb; my %wlpnpb;

my @wordocpb=(1707,2998,17877,35478,37713,40333,40699,48488,52310,52744);
foreach my $word (@wordocpb){
    my $g = 'OC';
    my $h = 'PB';
    my $r = low_label($word,$g,$h,$numminus);
    my @r = @$r;
    $rwocpb{$r[1]} = $word;
    $wlocpb{$word} = $r[0];
}

#my @wordpboc=(1707,7479,10487,17877,20676,22038,48488,50038,52744);
#foreach my $word (@wordpboc){
#    print "PB->OC\n";
#    my $g = 'PB';
#    my $h = 'OC';
#    my $r = low_label($word,$g,$h,$numminus);
#    my @r = @$r;
#    $rwpboc{$r[1]} = $word;
#    $wlpboc{$word} = $r[0];
#}

#my @wordocpn=(10487,17877,20676,22038,37713,40333,40699,48488,50038,51332);
#foreach my $word (@wordocpn){
#    my $g = 'OC';
#    my $h = 'PN';
#    my $r = low_label($word,$g,$h,$num_minus);
#    my @r = @$r;
#    $rwocpn{$r[1]} = $word;
#    $wlocpn{$word} = $r[0];
#}

#my @wordpnoc=(7479,10487,22038,35478,48488,50038,51332,52310,52744);
#foreach my $word (@wordpnoc){
#    my $g = 'PN';
#    my $h = 'OC';
#    my $r = low_label($word,$g,$h,$num_minus);
#    my @r = @$r;
#    $rwpnoc{$r[1]} = $word;
#    $wlpnoc{$word} = $r[0];
#}

#my @wordpbpn=(7479,10487,17877,20676,35478,37713,50038);
#foreach my $word (@wordpbpn){
#    my $g = 'PB';
#    my $h = 'PN';
#    my $r = low_label($word,$g,$h,$num_minus);
#    my @r = @$r;
```

```
69 #     $rwpbpn{$r[1]} = $word;
70 #     $wlpbpn{$word} = $r[0];
71 #}
72
73 #my @wordpnpb=(10487,22038,35478,48488,50038,51332,52310);
74 #foreach my $word (@wordpbpn){
75 #     my $g = 'PN';
76 #     my $h = 'PB';
77 #     my $r = low_label($word,$g,$h,$num_minus);
78 #     my @r = @$r;
79 #     $rwpnpb{$r[1]} = $word;
80 #     $wlpnpb{$word} = $r[0];
81 #}
82
83 #追加したラベルのログ用ファイル
84 my $flog = "log/plus${num}";
85 open my $fhlog, ">", $flog
86     or die "Cannot open $flog: $!";
87
88 #信頼度が低い順に配列に格納
89
90 my @wocpb;  #単語
91 my @locpb;  #ラベル
92
93 foreach my $key ( sort { $a <=> $b } keys %rwocpb ){
94     my $w = $rwocpb{$key};
95     push(@wocpb, $w);
96     push(@locpb, $wlocpb{$w});
97 }
98
99 my $lw = $wocpb[0];  #最も信頼度が低いラベルとその単語
100 my $ll = $locpb[0];
101 my $g = 'OC';
102 my $h = 'PB';
103 label_plus($lw,$g,$h,$ll,$fhlog);
104
105 #my @wpboc;  #単 語
106 #my @lpboc;  #ラ ベ ル
107
108 #foreach my $key ( sort { $a <=> $b } keys %rwpboc ){
109 #     my $w = $rwpboc{$key};
110 #     push(@wpboc, $w);
111 #     push(@lpboc, $wlpboc{$w});
112 #}
113
114 #$lw = $wpboc[0];
115 #$ll = $lpboc[0];
116 #$g = 'PB';
117 #$h = 'OC';
118 #label_plus($lw,$g,$h,$ll,$fhlog);
```

```
119
120 #my @wocpn;  #単語
121 #my @locpn;  #ラベル
122
123 #foreach my $key ( sort { $a <=> $b } keys %rwocpn ){
124 #    my $w = $rwocpn{$key};
125 #    push(@wocpn, $w);
126 #    push(@locpn, $wlocpn{$w});
127 #}
128
129 #my $lw = $wocpn[0];
130 #my $ll = $locpn[0];
131 #$g = 'OC';
132 #$h = 'PN';
133 #label_plus($lw,$g,$h,$ll,$fhlog);
134
135 #my @wpnoc;  #単語
136 #my @lpnoc;  #ラベル
137
138 #foreach my $key ( sort { $a <=> $b } keys %rwpnoc ){
139 #    my $w = $rwpnoc{$key};
140 #    push(@wpnoc, $w);
141 #    push(@lpnoc, $wlpnoc{$w});
142 #}
143
144 #$lw = $wpnoc[0];
145 #$ll = $lpnoc[0];
146 #$g = 'PN';
147 #$h = 'OC';
148 #label_plus($lw,$g,$h,$ll,$fhlog);
149
150 #my @wpbpn;  #単語
151 #my @lpbpn;  #ラベル
152
153 #foreach my $key ( sort { $a <=> $b } keys %rwpbpn ){
154 #    my $w = $rwpbpn{$key};
155 #    push(@wpbpn, $w);
156 #    push(@lpbpn, $wlpbpn{$w});
157 #}
158
159 #$lw = $wpbpn[0];
160 #$ll = $lpbpn[0];
161 #$g = 'PB';
162 #$h = 'PN';
163 #label_plus($lw,$g,$h,$ll,$fhlog);
164
165 #my @wpnpb;  #単語
166 #my @lpnpb;  #ラベル
167
168 #foreach my $key ( sort { $a <=> $b } keys %rwpnpb ){
```

```
169 #     my $w = $rwpnpb{$key};
170 #     push(@wpnpb, $w);
171 #     push(@lpnpb, $wlpnpb{$w});
172 #}
173
174 #$lw = $wpnpb[0];
175 #$ll = $lpnpb[0];
176 #$g = 'PN';
177 #$h = 'PB';
178 #label_plus($lw,$g,$h,$ll,$fhlog);
179
180 close $fhlog;
181
182 sub low_label{
183
184     my($word,$g,$h,$n) = @_; #単語,訓練,学習
185
186     #テストデータのラベル
187     my $fts = "${word}/label_test_${h}";
188     open my $fhts, "<", $fts
189   or die "Cannot open $fts: $!";
190
191     #テスト結果の信頼度
192     my $frs = "${word}/result${g}${h}${word}${n}";
193     open my $fhrs, "<", $frs
194   or die "Cannot open $frs: $!";
195
196     #ヘッダを取る
197     my $rsh = <$fhrs>;
198
199     my %lines;
200
201     while(my $tline = <$fhts>){    #テストデータから1行読み取り
202
203   chomp $tline;
204   $tline = decode('UTF-8', $tline);
205
206   my $rline = <$fhrs>;       #信頼度のデータから1行読み取り
207
208   $rline = decode('UTF-8', $rline);
209
210   my @rates;
211
212   while( $rline =~ /\s((?:\d\.\d+|\d))/g ){     #読み込んだ行の信頼度を全て取り出す
213       if( $1 >= 0 && $1 <= 1 ){
214     push @rates, $1;
215       }else{
216     push @rates, 0;
217       }
218   }
```

```
219
220     $lines{max @rates} = $tline;      #最も高いものが学習結果の信頼度
221                                       #その値をキーとしてハッシュに格納
222
223       }
224
225       close $fhts;
226       close $fhrs;
227
228       my @lines2;
229       my @lines3;
230
231       #信頼度が低い順に並べ替え
232       foreach my $key ( sort { $a <=> $b } keys %lines ){
233     push(@lines2, $lines{$key});    #ラベル
234     push(@lines3, $key);            #信頼度
235       }
236
237       #信頼度が最も低いデータを取り出す
238       my @r;
239       $r[0] = shift @lines2;
240       $r[1] = shift @lines3;
241
242       my $r = \@r;
243       return $r;
244
245 }
246
247 sub label_plus{
248
249       my($word, $g, $h, $s, $fhlog) = @_;
250
251       #訓練データのラベル
252       my $ftc = "${word}/label_teach_${g}";
253       open my $fhtc, "<", $ftc
254     or die "Cannot open $ftc: $!";
255
256       my @svm;
257
258       while( my $line = <$fhtc> ){    #1行ずつ読み込み配列に格納
259
260     chomp $line;
261     $line = decode('UTF-8', $line);
262
263     push(@svm, $line);
264
265       }
266
267       close $fhtc;
268
```

```
269     #出力用ファイル
270     my $ftcp = "${word}/label_teach_${g}";
271     open my $fhtcp, ">", $ftcp
272   or die "Cannot open $ftcp: $!";
273
274     #読み込んだ訓練データを出力
275     foreach my $data (@svm){
276
277   $data = encode('UTF-8', $data);
278   print $fhtcp $data,"\n";
279
280     }
281
282     #信頼度が最も低いデータを出力
283     $s = encode('UTF-8', $s);
284     print $fhtcp $s,"\n";
285
286     #ログ用ファイルに出力"
287     my $outlog = "${g}${h} : ${word} : ${s}";
288     $outlog = encode('UTF-8', $outlog);
289     print $fhlog $outlog,"\n";
290
291 }
```

ソースコード 7　負の転移有りの単語全体で信頼度最低のデータを訓練データに追加するプログラム（OC と PB の領域適応の場合）