目次

図目次

表目次

謝辞

本論文を作成するにあたり、多大なるご指導をして下さった開放環境科学専攻教授の櫻井彰人先生に心より感謝申し上げます。また、適切で丁寧なご助言を下さった開放環境科学専攻教授の山口高平先生、開放環境科学専攻助手の飯島正先生、開放環境科学専攻助手の篠沢佳久先生にも心より感謝申し上げます。

そして研究を進めるにあたって手助けして下さった先輩、後輩、ともに切磋琢磨した同輩、生活面で支えてくれた家族に心から感謝します。

平成 18 年 2 月 3 日

第 1 章　序論

1.1　背景

パーソナルコンピュータの普及、および世界中に張り巡らされたインターネット回線利用の低価格化により、専門知識を持たない一般の人々が情報技術を手軽に利用することが可能になった。近年では閲覧可能な資源、利用可能なサービスが増えただけではなく、多くの人が自ら情報を発信することが容易になってきた。特に日記などを目的としている文字情報（テキストデータ）の増大は目まぐるしく、ブログ（web log の略）や SNS（Social Networking Site）などの新しい技術がその流れに拍車をかけている。個人に限ったことではなく企業や公共機関も独自にホームページやデータベースを所有しているが、それらの情報はテキストデータだけではなく、画像や動画など多岐に渡ることが多く、管理すべき情報の種類とその容量は格段に増大している。

情報は今もなお増え続けているがゆえに、世界中に散らばる情報の中から、欲しい情報を素早く見つけ、分析し、効率的に選択していくことは難しくなってきている。自らが所有する莫大な情報においても同様であり、上手に管理する必要性が高まっている。

ここで注目すべきは、テキストデータはもちろんのこと、画像や動画などその他のファイルの名前もまたテキストで記述されていることである。つまり必要な情報を探し出す処理では、それらすべてのテキストデータを対象にしている。よって、我々が利用しているパーソナルコンピュータが、テキストデータをより高度に検索・分析・選択できるようになれば、その技術は社会に大きく貢献し、広く使われるようになる可能性が高いと言えるであろう。

パーソナルコンピュータとインターネット回線利用の広がりはさらに、機械翻訳に代表される多言語間に跨る処理を行う機会も増やしている。それに伴い、外国語で記述された情報を母国語に自動変換する技術は、英語を中心としてさまざまな言語間で研究が行われている。

これらの需要に対応する 1 つのアプローチとして自然言語処理技術がある。

1.2　研究の目的

本論文では自然言語処理の中でも浅い意味解析（Shallow Semantic Parsing）という技術に焦点をあて、日本語文への深層格付与を、SVM を用いて行うことを目的とする。そして、より高い精度での解析を目指す。さらに得られた結果から、日本語文に対する浅い意味解析において SVM を使用する際にどのような特徴量を用い、どのように特徴量の範囲を設定することが有効であるかを調べる。

1.3　論文の構成

本論文の構成は以下の通りである。

第 1 章　本研究に至った背景、本研究の目的を述べる。また本論文の構成についても示す。

第 2 章　自然言語処理の概要を述べ、本研究が属する分野である浅い意味解析の解説をする。さらに浅い意味解析の既存研究やそこで使用されている技術であるサポートベクターマシンについて述べる。

第 3 章　本論文の提案を述べるとともに、未決のパラメータを決定するための予備的な実験を行った結果を示し考察を述べる。

第 4 章　提案をもとに行った実験と、その結果について述べる。

第 5 章　実験結果についての考察を述べ、既存研究の結果との比較を行う。

第 6 章　本論文の結論を述べる。

第 2 章　研究の位置づけ

言語には、コンピュータが理解するための機械語、人間が相互理解のために使用する自然言語、そしてそれら二つの言語の橋渡しとなるプログラミング言語がある。本来、人間がコンピュータに対してより高度で複雑な処理をさせるためには、プログラミング言語の知識が不可欠であった。自然言語処理とは、ユーザがプログラミング言語を介することなく自然言語を計算機への入力とし、コンピュータがあたかもそれらを理解したかのような振る舞いをさせるための技術の総称である。自然言語処理はユーザとコンピュータの相互理解を助ける技術であるとも言える。

この章は、2.1 で本研究が属する自然言語処理について紹介し、2.2 で自然言語処理における浅い意味解析という分野について紹介し、2.3 で浅い意味解析の対象となる言語資料を紹介し、2.4 で浅い意味解析に関する既存の研究を紹介し、2.5 で浅い意味解析において有効なツールであるサポートベクターマシンの基本技術を説明する。

2.1　自然言語処理

2.1.1　概要

自然言語処理が研究され始めた初期の段階では、ルールベースによる解析的なアプローチが行われていた。これは、コンピュータが遭遇するあらゆるケースを解析し、それぞれにおいて行うべき処理をあらかじめ決めておき、ユーザの入力から推論を深めることで目的を満たすことを試みた方法である。この方法は、ある決められた特定の目的で使われる際に効果を発揮した。より解析を深め、より細かなルールを構成することで精度の向上も期待できる。しかしこの方法は、構築されたシステムが想定された状況でしか使用できないため、汎用性に非常に乏しい。さらに未知の入力がある度に新たなルールを追加する必要が出てくる。そのような際には、今まで構築してきたルールとの整合性に注意しながら慎重に作業を行わなければならない。ルールの規模が大きくなるとその作業は困難を極め、管理するのが難しくなる。

それらの問題を解決するために、1990 年代に入ってからは用例に基づくアプローチや統計的モデルに基づくアプローチが提案された[1]。用例に基づく方法は、大量の言語データを取得した上で、入力と類似性の高いデータを抽出し、それらを選択・組み合わせする事によって解析をするものである。統計的モデルに基づく方法もまた大量の言語データを用いるが、この方法では言語データから単語の出現頻度などの情報を抽出して統計的なモデルを構築する。これらの方法は未知の単語に対する処理に強く、新しい言語データの追加が容易であることが知られている。しかしより高い精度を求めるためには、良質で膨大な

言語データが必要になるという欠点がある。言語データの作成には、莫大な人数と手間をかけなければならないが、より多くの人、より長い時間をかけるとヒューマンエラーや人による判断の違いにより言語データの品質は低下する。これらの作業を、コンピュータを使うことで自動的に作成しようと試みる研究も存在するが、有効な技術は未だない。

現在もなお用例に基づくアプローチや統計的モデルに基づくアプローチが主流であり、それらを用いて研究が行われている。

2.1.2 二つの流れ（文章と文）

自然言語処理は､処理の対象とする文字列の長さにより 2 種類に大別することができる。1 つは文頭から句点までの「文」を対象とする処理であり、もう 1 つは複数の文の集まりである「文章」を対象とする処理である。

文を対象とする処理には、語の最小単位である形態素の区切りを解析することや、形態素同士が構文法に基づいてどのような関係性を持っているかを明らかにすること、文中で重要な役割を果たすとされる語にタグを付けることなどがあり､1 つの文の構造を解析することを目的としている。文章を対象とする処理には、文と文の論理的関係を明らかにすることや、代名詞や指示詞が何を表しているかを解決すること、省略された表現を補完することなどを行い、文章全体の構造を解析することを目的としている。

2.1.3 文に対する処理の流れ

本研究での対象でもある、文に対する処理に関する一般的な技術についてより詳しく説明する。

2.1.3.1 形態素解析

文に対する自然言語処理において最初に行われるのが形態素解析である。

言語において、意味や機能を成す最小単位の要素のことを形態素（morpheme）という。形態素解析には 2 つの段階がある。まず入力された文を形態素に分解し、次に活用変化などにおける語形の変化を解析して各形態素の原型を求める。英語など分かち書き[1]の習慣がある言語においては形態素の区切りが明確なので特に 2 つ目の段階のことを形態素解析と呼び、品詞（part of speech）の解析を行うことを主な処理とする。この処理を品詞タグ付け（part of speech tagging）と呼ぶこともある。これらの分かち書きをする言語では、英語の”like”のように名詞、動詞、前置詞、接続詞、形容詞といった複数の品詞をとることができる多品詞語が多く存在する。これら多品詞語に対する処理が分かち書きをする言語における形態素解析の難しさのひとつといえる。

[1] 単語と単語の間に空白を入れること．形態素の区切りは空白の認識で自ずと判定することができる．

これに対して分かち書きをしない習慣を持つ言語は、先の 2 つの段階を通して品詞タグ付けを行うことを形態素解析という。特に日本語における形態素解析について言及すると、狭義の意では最初の段階である形態素に分解することをいい、広義の意では分割した形態素の品詞、読みがな、アクセントなどの情報を解析することも形態素解析と呼ぶこともある。日本語の形態素解析では特に形態素の分割が困難とされている。それを示すものとして、「ここではきものをぬぐ。」という例文がよく挙げられる。この文の形態素区切りの結果は、

ここで｜はきもの｜を｜ぬぐ｜。　→　ここで履物を脱ぐ。
ここで｜は｜きもの｜を｜ぬぐ｜。　→　ここでは着物を脱ぐ。

という 2 つが存在するが、どちらの解析結果も構文的な誤りがない。このように複数の正しい結果が出力として得られることが形態素の分割を複雑にする。現状では、解析の前や途中で区切りを見つけるヒントを与えることや、最終的に得られた複数の結果から想定した正解を選択するなどといったユーザの手間をかける必要がある。また、

日本人 → 日本 ＋ 人
統計モデル → 統計 ＋ モデル
走り続ける → 走る ＋ 続ける

などの例に見られるように、どこまでを形態素とするかということも形態素の分割が困難となる原因のひとつである。意味や機能を成す最小単位をどのように定義するかは未だ議論の対象であり、すべての人が納得するような定義が存在しないのが現状である。

日本語の形態素解析の場合、表層の記述から読みがなを解析することも重要になることがある。例えば、

上手 ： ジョウズ／カミテ／ウワテ
正しく ： タダシク／マサシク

などの例で見られるように、同じ表層でも複数の違う読みがなを持つことがある。読みがなの解析は形態素を分割した後に行うことが多いが、読みがなの解析に誤りがあった場合には、以後の構文解析や意味解析に大きな影響を与える可能性がある。

2.1.3.2　構文解析

文の中の単語は一般的に、複数集まって一つの句を形成したり、単語や句が別の単語や

句を修飾したりする。文におけるこれらの情報を文法に則って解析することを構文解析（syntactic parsing[2]）という。構文解析の結果は木構造で表されることが多く、この木構造のことを構文木という。また構文解析を行う処理系のことをパーザーと呼んでいる。

一般的に文の構造分析には句構造規則を用いる。句構造規則の例を図 1 に示す。

A) 文 → 後置詞句 ＋ 動詞句
B) 動詞句 → 後置詞句 ＋ 動詞句
C) 動詞句 → 副詞 ＋ 動詞
D) 動詞句 → 動詞句 ＋ 助詞
E) 後置詞句 → 名詞 ＋ 助詞
F) 名詞句 → 形容詞 ＋ 名詞

図 1: 日本語の句構造規則の例

この規則を用い、文の構成要素を分析する。例えば、「彼は駅に到着した」という文においては、「彼」と「は」が規則 E により後置詞句を構成し、同様にして「駅」と「に」も後置詞句を構成する。「到着し」と「た」は規則 D により動詞句を構成するので、既に得た後置詞句と合わせて規則 A により文を形成する、といった分析を行う。得られた解析結果を木構造にすると、図 2 のような構文木になる。

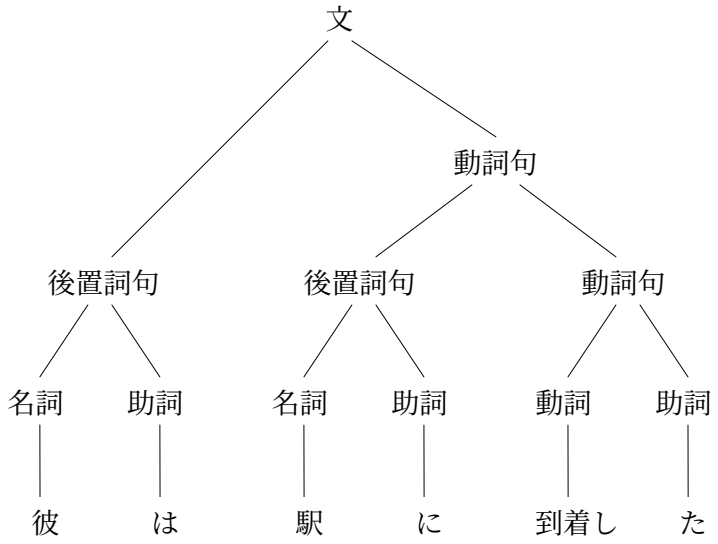


図 2: 「彼は駅に到着した」の構文木の例

句構造規則にはその制約の強さによって 4 つに分類される。形式言語理論によると、0 型文法、文脈依存文法（1 型文法）、文脈自由文法（2 型文法）、正規文法（3 型文法）である

[2] 単に parsing というときもある

[2]。型の数字は大きくなるほどに文法記述に関する制限が強くなる。4 型文法である正規文法は効率のよい解析手法があるが、文の階層構造などを十分に記述することができない。1 型文法である文脈依存文法は制限が緩む分だけ非常に広い範囲の言語構造を記述することができるが、効率のよい解析手法は存在しない。よって、文法記述にある程度の柔軟性を持ち、かつ効率のよい解析手法である 2 型の文脈自由文法が実用的にも広く用いられている。

文脈自由文法は、初期状態と 3 つの集合からなる。

$$G =< V_N, V_T, P, \sigma >$$

ここで、V_N は非終端記号の集合、V_T は終端記号の集合、P は生成規則の集合であり、σ は初期状態である。具体的に、非終端記号は「名詞句」や「動詞句」などの文法的なカテゴリーを表し、終端記号は「彼」や「到着する」などといった単語そのものを表す。左辺に初期状態σをもつ句構造規則から開始し、句構造規則を繰り返しその右辺に適用することで文を生成することができる。

文脈自由文法では、書き換えの規則が、

$$A \to \beta \ (A \in V_N, \beta \in (V_N \cup V_T)+)$$

の形をしている。簡単な例を挙げて説明する。

$$G =< V_N, V_T, P, \sigma >$$
$$V_N = \{a\}$$
$$V_T = \{a, b\}$$
$$P = \{\sigma \to a\ \sigma\ b,\ \sigma \to ab\}$$

このとき、非終端記号に対して生成規則Pを繰り返し適用して書き換えていくと、

$$\sigma \Rightarrow a\ \sigma\ b \Rightarrow a(a\ \sigma\ b)b \Rightarrow a_2(a\ \sigma\ b)b_2 \Rightarrow \cdots \Rightarrow a^{n-1}(a\ \sigma\ b)b^{n-1}$$
$$\Rightarrow a^{n-1}b^{n-1}$$

となり、これより文法Gは$a^{n-1}b^{n-1}$を表現することができる文法であるといえる。

つまり、与えられた文がどのような生成規則に従って生成されたかを解析することで、その文の構文情報を得ることができる。よって構文解析は、入力に対してその文を初期状態まで導くことができるような生成規則列を得ることであるということができる。

2.1.3.3　意味解析

意味解析とは、与えられた文からそれぞれの語の意味を明らかにし、文中に存在する別の単語との意味的な関係を解析することで文の意味構造を生成することをいう。意味構造は単語が持つ意味と、語や句、節などのまとまりの間に存在する意味的な関係を表したものである。意味構造の表現方法としては、解析の方法により木構造やネットワーク構造、フレーム表現などが用いられている。

意味構造を解析するための文法として代表的なものに格文法がある。

文の中で、単語や単語の集まりである句は一定の役割を担っている。格文法では、特に

述語が重要な役割をもっているとしている。また、文中の単語や句において意味的な役割を果たすものを格要素といい、その役割のことを格という。述語と格要素との関係を格関係と呼び、文には格関係で示される意味構造が存在するという考え方が格文法である。

ここで「太郎が花子に本を貸す」という例文を挙げると、格文法では述語である「貸す」を中心として、「太郎」、「花子」、「本」との間に格関係が存在すると説明する。図 3 は「太郎」は「貸す」に対する動作主格、「本」は目的格、「花子」は本を貸す対象格であることを示している。

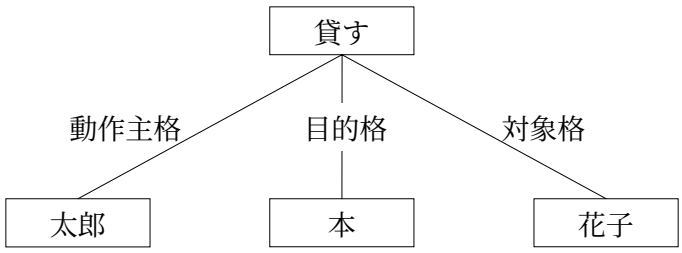


図 3: 格構造の表現例

また、格の中でもより構文的な性質を持つものを表層格、より意味的な性質を持つものを深層格という。深層格の概念を中心とする格文法の理論は、Charles J. Fillmore によって 1968 年に提案された。表 1 に Fillmore の深層格を示す。

表 1: Fillmore による深層格システム

深層格	説明
動作主格	動作を引き起こす主体
経験者格	心理的影響などを受ける実体
道具格	動作や状態の物理的な原因
対象格	動作の影響を受ける実体
源泉格	移動の開始点
目標格	移動の終了点
場所格	動作の空間的な位置や方向
時間格	動作や状態が起こる時刻

深層格は言語普遍である。すなわち、どの言語にも同じものが存在すると考えられている。

また、格文法により解析した結果をフレーム形式で表現したものを格フレームという。各フレームの例を図 4 に示す。

[“buy”(v.) [core [Buyer: NP, PP(by)] [Goods: NP, ADP]] [non-core [Money: (for)] [Place: (at),(in)]]]	[“買う”(動詞) [動作主格: ガ格] [目標格: ニ格] [対象格: ヲ格]]

図 4: 英語と日本語における格フレームの記述例

ある動詞が文中に出現したときに、その動詞によって定義された格フレームを参照することで、その文がとりうる意味的構造を知ることができる。この方法もまた言語依存性がない。

格フレームから意味構造を抽出する手順としては、まず構文解析の結果などから文中で述語となる語を知ることから始まる。述語が決定したら、あらかじめ用意された格フレームの集合から該当する述語に対応した格フレームを抽出し、その格フレームがとりうる格とその制約条件より格要素となる語を文中より検索する。一致するものがあればその単語もしくは句を格要素とする。

2.1.4 応用技術

自然言語処理の技術が発展すると幅広い分野でその恩恵を受けることになる。まず考えられるのは検索技術への応用である。近年、インターネットの検索サイトを運営する会社が様々なサービスを立ち上げて注目を浴びているように、検索の分野が発展することによる社会への影響力は非常に大きい。それらのサービスの集約とも言える画像や動画を含めたサイト検索では、現在キーワード検索が主流となっている。構文解析、意味解析の精度向上、処理の効率化、ユーザが不快に感じない時間での解析が実現した場合には、自然言語による検索が実現する可能性がある。これにより、キーワード検索では実現できなかった、より人間の直感に近い形での表現での高度な検索が可能になり、計算機とユーザとの親和性も飛躍的に向上すると考えられる。それらがまた新しいサービス、新しい市場を作るきっかけにもなるかもしれない。同様に、質問応答の分野への応用も可能である。ユーザの自然言語での問いかけに対してその意味的な真意を汲み取り、適切に判断された的確な言葉によって応答をするシステムが作られるかもしれない。

また、各言語における自然言語処理技術が進展し、さらに各言語間の性質の違いやすべての言語に共通する性質などが明らかになれば、機械翻訳の分野への応用が可能となる。ある言語が持つ特有の表現を別の言語のより近い表現に変換できたり、文法的に不完全な砕けた文に対してもより精度の高い翻訳ができたりすることが望ましい。さらには音声認識の分野との協調により、異なる言語を話す人同士がオンラインでの会話を楽しめるようになることも考えられる。

自然言語処理を自動化するには、これまで説明した形態素解析、構文解析、意味解析のすべての処理を自動化しなければならない。一般的に、自動化する度合いが高い分だけ精度が下がってしまうが、結果を次の処理で使用するシステムにおいては、ひとつの処理の精度低下はその後の処理に大きな影響を与えてしまう。つまり上記の応用技術が現実となるためには、それぞれの処理の自動化をより正確に行う必要がある。

それぞれの解析の精度向上もまた、他技術への応用が可能である。例えば、形態素解析の精度が上がり文字の区切りが正確に分析できるようになれば、私達が使っているパーソナルコンピュータで文章などを作成する際の文字変換技術が向上するであろう。

2.2 浅い意味分析（Shallow Semantic Parsing）

浅い意味解析は、文の述語動詞に注目して格要素を抽出する解析のひとつである。その概要と、浅い意味解析を行う意義やその応用に関して述べる。

2.2.1 位置づけと概要

浅い意味解析は、解析する文における述語動詞に注目してその他の単語との格関係を解析する処理のことであり、タスクは 2.1.3.3 で紹介した意味解析と同様である。意味解析と浅い意味解析との相違点は、求める格要素の種類にある。一般的に意味解析では数多くの格のセットが用意されている。FrameNet プロジェクト（2.3.1 参照）を例にとると、合計数百もの格が用意されており、単語の役割をより詳しく分類しようとしている。それに対して浅い意味解析は、「いつ」、「どこで」、「どのように」、「誰が」、「何を」、「誰に」、といった比較的単純な役割を解析する[5]。より”浅い”概念の格を割り振る分析が、浅い意味解析であるといえる。

浅い意味解析における単語への役割付与の一般的な処理として、識別（identification）と分類（classification）の 2 つに分けることができる。識別とは、文の中から格要素となる単語および句を決定する処理である。識別の結果、すべての句、単語は文中で文の主要述語に対し意味的な役割を持つか持たないかに分かれることになる。そして、役割を持つと判断されたものに対して具体的にどのような格を持つかを決定する処理が分類である。「太郎は花子に本を貸した」を例にして浅い意味解析を行った結果を図 5 に示す。ただし、

ここでは格要素の単位は単語であるとする。

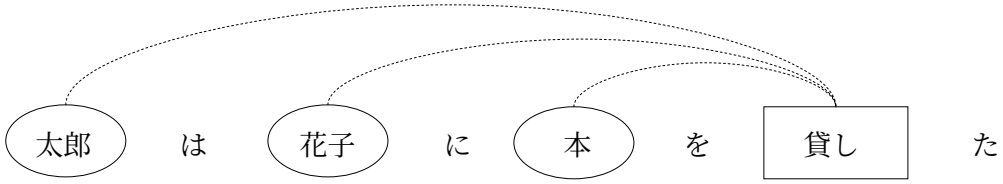


図 5: 浅い意味解析における識別

この段階では「貸し」という述語に対し、「太郎」、「花子」、「本」が何らかの役割を持つことが決定されているが、実際にどのような役割を持つかは決定されていない。さらに分類を行った結果を図 6 に示す。

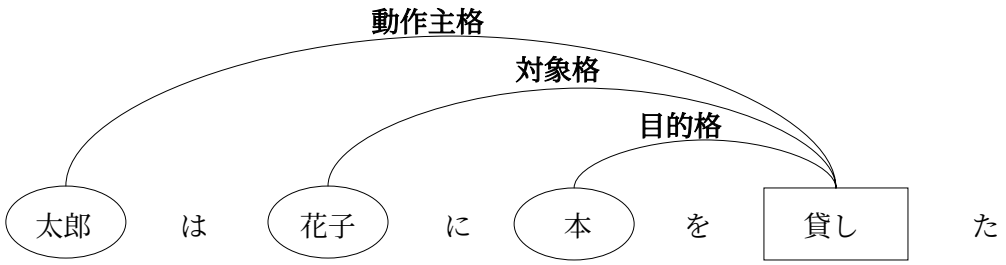


図 6: 浅い意味解析における分類

図 6 は識別処理によって役割を持つとされた単語に対して、あらかじめ用意された格のセットの中から該当する格を付与した結果である。この 2 つの処理により、浅い意味解析を実現する。

2.2.2 浅い意味解析の必要性

浅い意味解析では単語を認識するために形態素の区切り位置が必要となる。よって形態素解析はあらかじめ行う必要があるが、単語単位で格を付与する処理の場合には句の構成情報を必ずしも知る必要がないため、構文解析を介さずに文の意味的な解析を行うことができる。一般的に、構文解析は多大な計算量を要するので、これを省いて処理ができることが浅い意味解析の利点の一つである。より少ない情報から有効な情報を速く抽出することができれば検索技術などへの応用が期待できるため、浅い意味解析の技術向上が必要とされる。

さらに、浅い意味解析で使用する格のセットの中でも表 1 で示した Fillmore の深層格は

言語普遍と考えられるので、それらの格の付与を目的とした研究が世界に広がった場合には、各言語での研究結果を共有しやすい環境が構築されるであろう。それらは機械翻訳などの分野に貢献する十分な可能性を秘めている。

よって浅い意味解析は非常に重要な研究分野であるといえる。

2.3 言語資料（コーパス）

コーパスとは、大量の言語データのことをいう。最近では何らかの解析がされたコーパスの需要が高いので、そのニーズを満たすコーパスが盛んに作られている。本論文ではそのようにコンピュータで処理することを目的としてさまざまな解析結果が付与されている大量の言語データのことをコーパスと呼ぶ。コーパスには格要素の意味を示す格の名前とその定義、および格のセットの種類や数がそれぞれ定められている。現在広く使われている代表的なコーパス、およびそのプロジェクトを以下に紹介する。

2.3.1 FrameNet Project[3]

FrameNet Project は、カリフォルニア大学バークレー校の Fillmore を中心として行われている取り組みであり、Fillmore 自身が提唱したフレーム意味論の枠組みに則ってコーパスを作成している。フレーム意味論では、話者がある語の意味を理解するにはその語の認知的な背景基盤（フレーム）を考慮する必要があると提案している。その提案に基づき、語彙に関するネイティブスピーカーの知識を語彙項目ごとに定義し、豊富な例文をつけて整理されたものが FrameNet コーパスである。

例えば「bachelor」は「結婚していない、大人の男性」という意味で通常使われるが、この定義を満たしていても「bachelor」と呼ぶことがふさわしくない例が存在する。例として、法王はそもそも結婚することがないという前提があるために、未だ結婚をしていないという意味を持つ「bachelor」と表現するのは明らかに不自然である。この場合、結婚についての社会制度・文化概念等の背景知識を無視して意味解析を行うと正しい結果が得られないことになる。またもうひとつの例として「ground」と「land」という 2 つの単語を挙げる。これらはともに「大地」を指示している。ここで、

a) the traveler took a rest on the ground
b) the traveler took a rest on the land

という 2 つの文があったとき、ここでも 2 つの単語はともに「大地」を意味するが、「ground」は「空」と対になる「大地」、「land」は「海」と対になる「大地」を意味する。このようにそれぞれの単語が異なる背景を想起させるため、a)は空の旅の中断であり、b)は海の旅の中断であると推測できる。フレームはこのような問題を解決するために用いている。

[3] http://framenet.icsi.berkeley.edu/

FrameNet では、ある単語がさまざまな意味で使用されている場合には、その語彙項目を lexical unit という単位で明示している。また各フレームは、フレームの要素である frame element の範囲が決められており、この frame element が FrameNet における格の表現となる。

FrameNet のデータベースには、

- frame、frame element、lexical unit の定義
- 各 lexical unit の結合化パターン
- lexical unit ごとに分類され、frame と frame element が明示された例文

が記述されている。

2.3.2 PropBank Project[4]

PropBank Project は、ペンシルバニア大学で行われている取り組みである。構文解析を目的としている Penn Treebank コーパスに対して意味解析を行い、その結果を付与したものである。PropBank Project では単語が文中で持つ意味のことを argument と呼んでいる。格の名前は FrameNet のように意味ごとに付けられているのではなく、付与される深層格は Arg0 から Arg5 と決められている。そして Arg0 から Arg5 は、動詞ごとにその定義が示されている。一般的に Arg0 は文の中での主語、Arg1 が述語に対する第一目的語、Arg2 が第二目的語であることが多い。PropBank コーパスの例を図 7 に示す。

Roleset rent.01 "be a renter":	John rented a room from Mary for a year, then moved out.
Roles: Arg0:renter Arg1:thing rented (eg, apartment) Arg2:landlord Arg3:rent	Roles: Arg0:John REL: rented Arg1:a room Arg2-from:Mary Arg4-for:a year

図 7: 動詞“rent”の主語が借りる人である場合の argument の定義とコーパスの例

図 7 の例では、「rented」という述語に対して、「John」が Arg0 となり、「a room」という句が Arg1、「Mary」が from という前置詞を伴って Arg2、「a year」が for を伴って Arg4 となり、さらにこの文には Arg3 が存在しないことも示している。

[4] http://www.cis.upenn.edu/~mpalmer/project_pages/ACE.htm

研究によっては、文の中で必ず必要になる主語や目的語に付与される argument を core argument、必ずしも必要とはいえない副詞や副詞句を adjunctiveargument として区別することもある。この場合、core argument は Arg0 から Arg5 に、adjunctive argument は ArgM-TMP（時間）や ArgM-LOC（場所）などといった名称が argument として付与される。

2.3.3 EDR 電子化辞書[5]

EDR 電子化辞書は、知的情報処理のソフトインフラ開発を目的として、1986 年度から 1994 年度の間に基盤技術研究促進センターとコンピュータメーカー8 社の出資により行われたプロジェクトの成果物である。EDR 電子化辞書には単語辞書、対訳辞書、概念辞書、共起辞書、専門用語辞書、EDR コーパスがあるが、本研究では EDR コーパスを使用するので日本語の EDR コーパスについてのみ言及する。

EDR コーパスは約 20 万文の文例があり、それぞれの文例情報はレコード番号、文情報、構成要素情報、形態素情報、構文情報、意味情報、および管理情報から構成される。

形態素情報は、図 8 のようになっている。

<構成要素番号>	<表記>	<かな表記>	<品詞>	<概念選択>
1	部屋	ヘヤ	名詞	3c0841
2	の	ノ	助詞	2621d5
3	電気	デンキ	名詞	102ab4
4	が	ガ	助詞	2621d5
5	明る	アカル	動詞	3ce654
・				
・				
・				

図 8: EDR コーパスの形態素情報

構成要素番号とは、文頭から文末までの各形態素に対してつけた昇順の番号のことである。概念選択とは、その形態素がどのような意味で使用されているのか、その概念を示すものである。

次に意味解析情報を図 9 に示す。

[5] http://www2.nict.go.jp/kk/e416/EDR/

例文:このため、媒体の価格も徐々に下がり始めている。

```
[       [main 9:下が:3ce9da]
        [attribute progress begin]
        [object [
                [main 6:価格:3cebde]
                [modifier 4:媒体:10495e]
               ]
        ]
        [manner 8:徐々に:0f81ac]
        [condition 2:ため:0e84ad]
]
```

図 9: EDR コーパスの意味解析情報

意味解析情報はフレーム表現で記述されている。全体の表現は、文の述語となっている概念を示す記述「main」から始まり、その概念と関係している他の概念について関係を示した表現と、概念の属性を表す表現が続く。main との関係や他の概念との関係示すものを概念関係子という。概念関係子には、動作主(agent)、対象(object)、道具(implement)といったコト概念からモノ概念へ向かう関係子や、条件(condition）や連続事象(sequence)などのコト概念同士を関係づける関係子や、所有関係(possessor)などの仮関係子がある。（付録 1 参照）

また文の時制や相の情報を表すものにS-attribute、attributeなどがありこれを概念属性子という。（付録2～5参照）S-attributeに文全体の内容に対する話者の視点、attributeには文の要素に対する話者の視点などが概念属性子で表現されている。また、特に特殊なものとしてwhichが挙げられるが、これは名詞を修飾している埋め込み文などを記述するために用いる。埋め込み文は体言に対応する概念をmainとして、図 10のように表現する。

例：彼が書いた字

```
[     [main 6:字:3d0797]
      [which [
            [main 3:書:0e910d]
            [agent 1:彼:2dc304]
            [object 6:字:3d0797]
            [attribute already end]
      ]]
]
```

図 10: which を使った埋め込み分の表現方法

EDR 日本語コーパスでは、これらのレコード番号、文情報、構成要素情報、形態素情報、構文情報、意味情報、および管理情報が 1 文につき 1 行で表現されている。

EDR 日本語コーパスは約 20 万文という規模の大きさに非常に大きな利用価値がある。しかし、このコーパスの作成には非常に多くの人が長い間関わっているために、形態素の区切り方、品詞を決定する基準、構文の構成基準、各深層格の付与基準などの曖昧さが存在し、人間の手で作られたことによる決定基準の歪みがあることが否めない。つまり、表層的には正しくても付与された情報の歪みにより正しく処理できない文が存在する可能性があると言える。

また、文の述語に限らず意味的なつながりを持つ単語の関係を詳細に記述している半面、modifier や a-object など定義の曖昧な関係概念子が大量に存在している。これらの概念関係子は機械学習の精度を低下させる恐れがある。さらに浅い意味解析には述語とその他の単語の関係を表す関係子だけでよいので、EDR コーパスを浅い意味解析で利用する際には、使用する概念関係子、概念属性子を吟味する必要がある。

2.4 既存研究

浅い意味解析を SVM を用いて行った研究を、浅い意味解析を日本語コーパスに対して行った研究を紹介する。

2.4.1 SVM を用いた浅い意味解析

SVM を使用して意味解析を行った研究の中でも、Sameer Pradhan[5][6]（PWHMJ2004）はこの分野の中でも最も高い成果を上げた研究の 1 つである。PWHMJ2004 は PropBank の英語コーパスに含まれる文を入力とし、Charniak Parser を用いて構文解析を行った後

に SVM で使用するための特徴量を抽出して、SVM を使った意味解析を行うという手法を用いた。意味タグ付けには YamCha と TinySVM を用いている。手法のモデルを図 11 に示す。

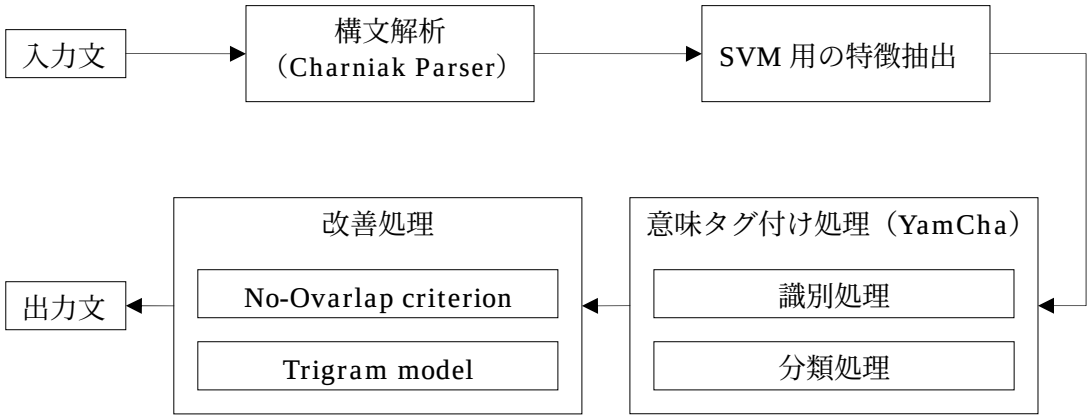


図 11: PWHMJ2004 の処理の流れ

argument は core argument と adjunctive argument に分けられて付与され、すべての argument に対する精度と core argument のみの精度とに分けて結果が出されている。また、SVM の入力としてコーパスから抽出する特徴量を表 2 に示す。

表 2: PWHMJ2004 が使用した SVM の特徴量

特徴量の名称	説明	例, 補足
Predicate	文の述語	rent, talk
Phrase Type	対象となる句の品詞	NP(名詞句), VP(動詞句)
Path	対象の句から Predicate までの構文木上の通り道を表したもの	NP↑S↓VP↓VBD
Position	対象の句が Predicate の前後どちらかにあるかを表す 2 値の特徴量	0, 1 のみ
Voice	文が能動態か受動態かを表す 2 値の特徴量	0, 1 のみ
Head Word	対象の句の先頭にある単語	—
Sub-categorization	Predicate の親となる句がどのように展開されているか	VP↓VBD
Named Entities in Constitutes	名詞句に対して 7 種類のタグを付与したもの	Person, Time, Location, Organization, Percent, Money,
Head Word POS	Head Word の品詞	—
Verb Clustering	Predicate の動詞を 64 種類に分類した結果	—
Partial Path	対象の句と, predicate の共通親ノードまでの Path	—
Verb Sense Information	動詞がどの Lexical Unit として使われているかを表したもの	—
Head Word of Prepositional Phrase	前置詞句の記号に前置詞を加えたもの	at the park → PP-at
First and Last Word/ POS in Constituent	対象の句の最初と最後の単語とその品詞の計 4 つを含む特徴量	—
Ordinal Constituent position	predicate から対象の句までの標準的な要素の並び順を表したもの	—
Constituent tree distance	対象の句の周りの状況を表す 9 種類の特徴量	親ノード,左右の句の品詞,head word とその品詞
Temporal cue words	時制を表す単語かどうかを示す 2 値の特徴量	0,1
Dynamic class context	解析中に付与された argument を 2 つ保持したもの	—

図 11 のシステムを改善するため No-Ovarlap criterion と称する改善策がなされている。これは「argument の範囲は別の argument の範囲と重なることはない」という制約を加えて

解析を行うものである。もう 1 つの改善策である trigram model は、3-gram によって、argument の候補から一番確率の高い並び方を選択することで argument の決定を行っていくというものである。ただし、Adjunctive Argument は副詞句が選択されることが多いために一般的に語順による制約を受けないことから、Adjunctive Argument には Trigram model は適用せず、Core Argument のみを対象とする。

実験は、2002 年 7 月公開の PropBank データ Section02-21 のうちの 51,000 文で学習、Section-23 の 2,700 文でテストを行ったものと、2004 年 2 月に公開された PropBank データ Section02-21 の 85,000 文で学習し、Section-23 の 5,000 文でテストを行ったものと分かれており、別々に結果を出している。結果は表 3 の通りである。

表 3: PWHMJ2004 の結果

Classes	Task	適合率	網羅率	F1	Accuracy
ALL ARGs	Identification	0.893	0.829	0.86	—
	Classification	—	—	—	0.900
	Id. + Cl.	0.840	0.752	0.794	—
CORE ARGs	Identification	0.920	0.833	0.874	—
	Classification	—	—	—	0.905
	Id. + Cl.	0.864	0.784	0.822	—

Core Argument に関しては、システムによる深層格付与の自動化を行っても F_1 が 8 割を超えている。この研究では、SVM が浅い意味解析において非常に有効なツールであることを示した。

2.4.2 日本語文に対する浅い意味解析

日本語文に対して浅い意味解析を行った研究として、渋木[7]の研究が挙げられる。渋木の研究では日本語の文例として EDR 日本語コーパスを用いている。処理の流れを図 12 に示す。

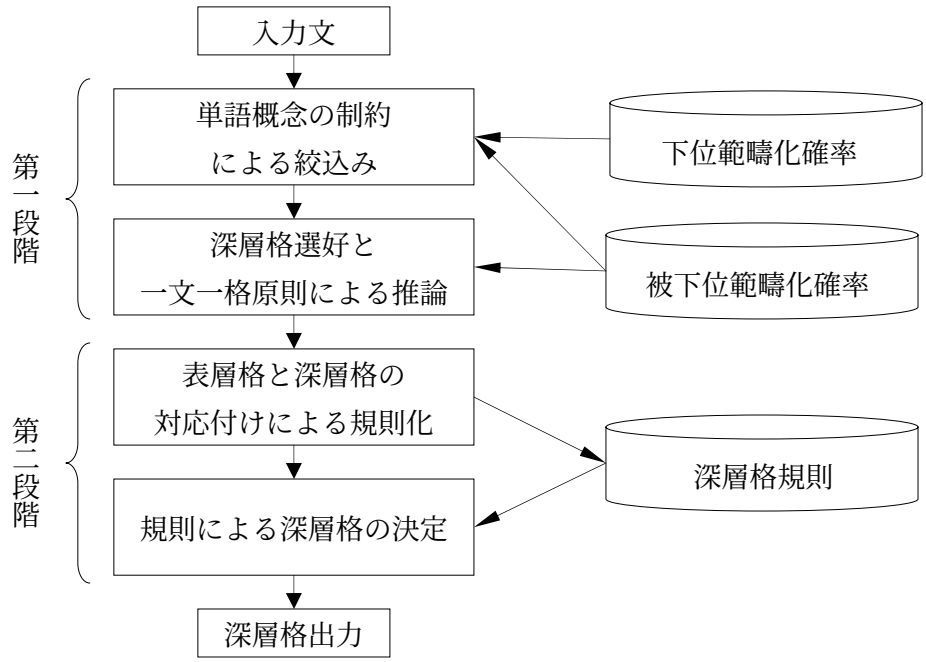


図 12: 渋木の処理の流れ

処理の手順は二段階に分かれているが、その準備として意味フレーム内での各単語の下位範疇化確率、被下位範疇化確率を求めている。下位範疇化（subcategorization）とは、述語が「飲む」なら、「どこで」や「何を」に該当するものをいい、文中のある動詞に対して下位範疇化されている場合、以下の式で下位範疇化確率を計算する。

$$P(c,d) = \frac{F(c,d)}{\sum_{k \in D} F(c,k)}$$

c は、主に 6 桁の 16 進数で表された概念情報のうちの上位 5 桁の概念情報のことであり、D は深層格の集合、d は特定の深層格、$F(c,d)$ は概念 c が深層格 d と共起した回数を表している。つまり $P(c,d)$ は、概念 c と深層格 d が共起する確率を計算し、これを被下位範疇化確率と定義している。被範疇化確率が 0 である概念 c と深層格 d は共起せず、0 以外のときは少なくとも一回は共起することを示しており、概念 c と深層格 d の選好の指標とすることができる。さらに渋木はこれらの情報を言語普遍の知識であると主張している。

あらかじめコーパスより計算された下位範疇化確率と被下位範疇化確率を用いて、第一段階である単語概念の制約による深層格の絞込みを行う。具体的には、被下位範疇確率もしくは下位範疇化確率が 0 である単語の関係を推測候補から除外して、係り元の動詞および係り先の単語概念の深層格関係を絞り込む。次に一文一格の原理に則り、さらに絞り込む。一文一格の原理とは、1 つの用言に同じ深層格持つ格要素が複数係ることはないという制約のことである。一文一格の原理による推測とは、まず範疇化確率から深層格の仮説を立て、一文一格の原理による矛盾が生まれないかを検証し、仮説を否定もしくは次の仮説

を立てていくことである。図 13 を用いてこの手順を説明する。

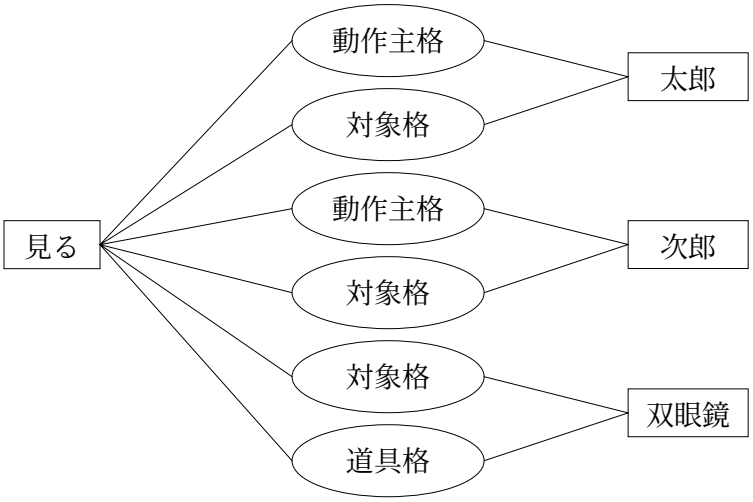


図 13: 渋木の研究における深層格推測第一段階の例

図 13 は、単語概念の制約による絞込み処理の結果である。ここで「見る」と他の単語を結ぶ関係のうち、「双眼鏡」が道具格である可能性が一番高いと仮定すると、「双眼鏡」が対象格であると仮定できる。これが正しいとすると、「太郎」と「次郎」は対象格でないと推測できる。この場合、「太郎」も「次郎」も対象格ではないために取り得る深層格は共に動作主格のみとなり、一文一格の原理に反する事になってしまう。よって最初の仮説は棄却され、「双眼鏡」が道具格であると推測できる。

処理の第二段階ではまず、深層格を一意に決定できる表層格[6]を探す。コーパスの中で出現する表層格のうち 7 割以上の割合で特定の深層格を取り得るものを、深層格を一意に決定する表層格とする。第一段階で得られた推測と異なる深層格が推測された場合は、第二段階の結果を出力する。

これらの手順に従って付与する深層格は、agent、object、cause、material、source、goal、place、purpose、basis、beneficiary、quantity の 11 種類である。

この研究では、コーパスに対して制約を設け、処理の対象となる文を絞り込んでいる。処理の第一段階の始めの処理である単語概念の制約がそれに当たり、動詞とその他の単語の組を作る際にはそれらの単語が分類語彙表[7]にあることを前提としている。

渋木のシステムによる浅い意味解析の結果は表 4 の通りである。

[6] 表層格とは格要素末尾にある「が」「を」「に」「は」「では」などの助詞列を表す
[7] Appendix 参照

表 4: 渋木の結果

適合率	網羅率	正解率
0.540	0.928	0.501

網羅率は、第一段階においてすべての動詞とその他の単語についての共起確率を獲得しているために非常に高い。しかし適合率と正解率は 5 割程度であり、実用化には精度の向上が不可欠である。

2.5 サポートベクターマシン (SVM)

本研究では、浅い意味解析を実現する技術としてサポートベクターマシン(Support Vector Machines :SVM)を採用している。本節ではサポートベクターマシンについて説明する。

2.5.1 概要

サポートベクターマシンは、未知のデータを 2 種類のクラスに分類（2 値分類）するための識別器の１つである。この識別器を構成するためには、あらかじめ用意した教師データを用いて特徴空間を２つのクラスに分離させる境界を求めることが必要となる。この処理を学習または訓練と呼ぶ。

2.5.2 学習

学習は、既知の学習用データから特徴ベクトルを抽出し、その特徴ベクトルと分類されるクラスC_1, C_2との確率的な対応付けを行うことである。未知データを定められたクラスに分類するためには、まず分類する対象から何らかの特徴量を抽出しなければならない。一般的には複数の種類の特徴量を抽出することが多い。獲得された特徴量をまとめ、ここでは特徴ベクトル$\vec{x}^T = (x_1, \cdots, x_a)$として表現する。ここで$\vec{x}^T$は、ベクトル$\vec{x}$の転置行列を表し、a は特徴ベクトルの次数、つまり特徴量の種類の数を表している。サポートベクターマシンは図 14 のような線形の閾素子の構造をとっている[3]ことより、これら特徴ベクトルから識別関数、

$$y = \mathbf{sign}\left(\sum_{i=1}^{a} w_i x_i - b\right)$$

を用いてyを算出する。wは個々の入力に対する重み値、bは閾値である。

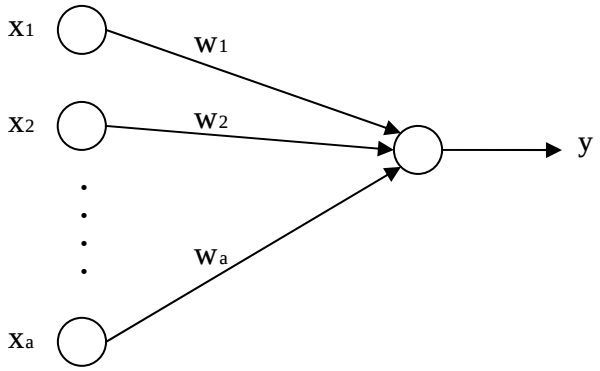


図 14: 線形の閾素子

y は右辺カッコ内の値が 0 以上のときに 1 をとり、0 より小さいときに−1 をとる。ここでクラス C_1, C_2 をそれぞれ $1, -1$ と数値的にラベル化する。学習用データでは特徴ベクトルと付与されているラベルが既知なので、特徴ベクトルおよび y で算出されるクラスラベルとの対応が極力一致するように、重み値 w と閾値 b を調節していくことで分類器を構成していく。これをより幾何学的に理解しようとすると、図 15 のような識別空間に配置された学習用データを、2 つのクラスを分類するための境界超平面を求めることに相当する。

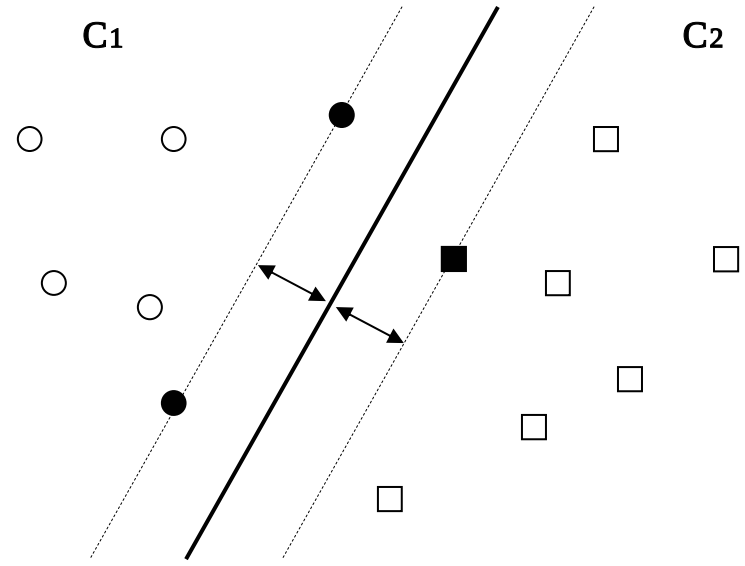


図 15: 識別平面におけるサポートベクターとマージン

境界超平面に近い学習サンプルを特にサポートベクターと呼ぶ。図 15 では○が C_1 のサンプル、□が C_2 のサンプル、●と■がサポートベクターを表しており、矢印はそのときのマージンであり、得られた最適分離線を実線で示している。

2.5.3　最大マージン化

学習用データを分離する境界平面は複数存在する。一般的に境界平面は一意に決まらないが、より余裕を持って分けられることが望ましい。そこで、境界平面に最も近い訓練サンプルとの距離をマージンと呼び、マージンが最大になるような境界平面を求める。

訓練用データの集合が完全に線形分離可能であると仮定すると、

$$t_i\left(w_i x_i - b\right) \geq 1, \qquad i = 1, \cdots, a$$

を満たすt_iが存在する。この条件下では$w_i x_i - b = 1$と$w_i x_i - b = -1$の 2 つの超平面で訓練用データは完全に分離されており、さらにこの 2 つの超平面の間に訓練サンプルは存在しない。このとき、これら超平面と識別平面との距離がマージンであり、その値は$\frac{1}{\|w\|}$となる。

よってマージンを最大化する問題は、制約条件

$$t_i\left(w_i x_i - b\right) \geq 1, \qquad i = 1, \cdots, a$$

の下で、

$$\min_w \|w\|^2$$

の問題を解くことに等しい。そしてこれを解くことによって、複数の境界平面の中から最適なものを選択することができる。

2.5.4　カーネルトリック

これまでは線形分離が可能であることを前提としたが、パターン識別を必要とされるデータの分類において線形分離できることは稀である。一般的に、線形分離ができる可能性は特徴ベクトルの次数が高くなるほど容易になり、訓練サンプル数が多くなるほど難しくなる。そこで特徴ベクトルを非線形化した高次元の空間に写像し、その空間で識別を行うことでサポートベクターマシンは線型分離を実現させた。一般に高次元空間に写像して分類器を構成すると汎化能力が低下する。マージン最大化はこの欠点を解消する方法である

マージンを最大化する問題においてラグランジュ乗数を用いる際、特徴ベクトルの内積計算に多大な計算量が必要になる。そこで、この内積の計算をより簡単な関数に置き換えて高次元空間での分類を容易にした。この置き換えの手法をカーネルトリックと呼んでいる。元の特徴ベクトルを$\vec{x}$、非線形への写像を$\phi(\vec{x})$とすると、カーネルKは、

$$\phi(\vec{x}_1)^T \phi(\vec{x}_2) = K\left(\vec{x}_1, \vec{x}_2\right)$$

で表される。カーネルKには例えば、多項式カーネル、

$$K\left(\vec{x}_1, \vec{x}_2\right) = \left(1 + \vec{x}_1^{\,T} \vec{x}_2\right)^p$$

Gauss カーネル

$$K(\vec{x}_1, \vec{x}_2) = \exp\left(\frac{-\|\vec{x}_1 - \vec{x}_2\|^2}{2\sigma^2}\right)$$

シグモイドカーネル

$$K(\vec{x}_1, \vec{x}_2) = \tanh\left(a \cdot \vec{x}_1{}^T \vec{x}_2 - b\right)^p$$

などが使われている。

2.5.5 多値分類への拡張

前述の通り、サポートベクターマシンは 2 値分類器であるために、単独で直接多値分類に応用することはできない。多値分類の実現は、複数のサポートベクターマシンを組み合わせることで可能となる。多値分類を目的とした複数のサポートベクターマシンを組み合わせる方法として代表的なものに、Pairwise 法と One-versus-All[8]法(OVA)という 2 つの方法がある。

2.5.5.1 Pairwise 法

データをn個のクラス$C_1, C_2, \cdots, C_n$に分類したいとき、Pairwise法ではすべてのクラスの組合せ、つまり${}_N C_2$個の分類器を学習の際に構成する。たとえばA, B, Cの3つのクラスに分類したい場合、A vs B、A vs C、B vs Cの3つの分類器ができることになる。未知データを分類する際には、そのすべての分類器に対して入力を行い、各クラスのスコアを計算して一番高かったクラスに分類される。

2.5.5.2 One-versus-All 法(OVA)

OVA とは、C_1とそれ以外を分ける分類器、C_2とそれ以外を分ける分類器、といった具合に、ある特定のクラスに属するものとそのクラスに属さないものとに分類する分類器を構成していく方法のことである。n 個のクラスに分割したいとき、OVA では n 個の分類器を構成することになる。分類の際には、すべての分類器に対して入力を行い、一番スコアの高かった分類器が分類しようとするクラスに属するという判断を行う。OVA のイメージを図 16 に示す。

[8] One-versus-Rest とも言う

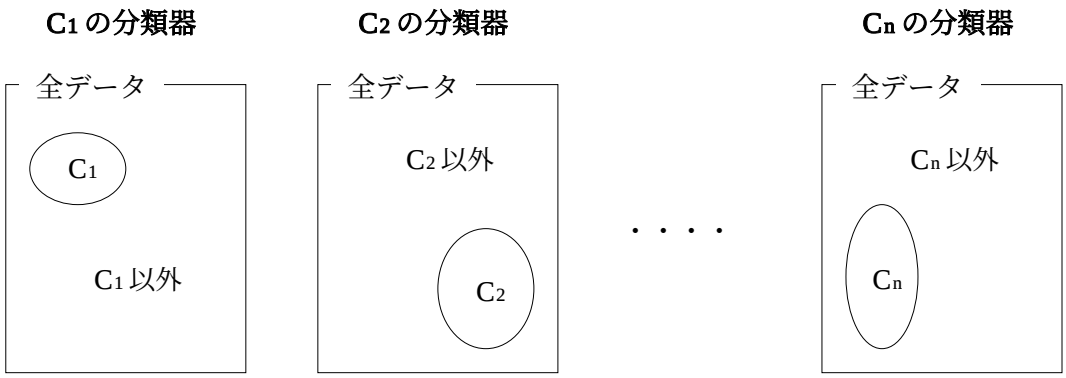


図 16: OVA の分類器構成のイメージ

2.5.6 短所と長所

サポートベクターマシンは存在する数多くの識別法の中でも最も優れた手法のうちの 1 つであると考えられている。第 1 に、未学習データに対して高い識別性能が得られるということである。マージン最大化や、特徴ベクトルの設定の仕方次第で精度の高いクラス分類が実現できる。第 2 に、ラベル付けされたデータを集めて共通のアルゴリズムを動かすだけで学習ができるため、非常に手軽であるということである。サポートベクターマシンを実装するに当たって既存のツールなどを用いれば、自ら設計、プログラミングをする必要はない。3 つ目の理由として、応用事例の範囲が広いことが挙げられる。パターン認識が必要なさまざまな分野でのパターン識別、クラスタリングなどに用いることができる。

しかしながら、学習の手順の中に十分な理解がしにくい部分が存在する。まずサポートベクターマシンは多数のパラメータを調節しているため、学習の経過が見えにくい。また、訓練サンプルが少ないと非常に汎用性の乏しい分類器しか構成することができない。分類すべきクラス数が大きい場合は、なお大規模な訓練サンプルが必要になる。しかし大規模な訓練サンプルがあったとしても、過学習をおこしてしまう原因となる場合がありえる。さらに、大域的な最小解を求めるといった作業には向かず、問題によっては非常に効率の悪い場合があることも短所の一つとして挙げられる。

これらを十分に分析した上でサポートベクターマシンを使用する必要がある。

2.5.7 タグ付けを実現するツール（TinySVM+YamCha）

SVM による分類を作成し、何らかのタグを付与する処理を実現するツールとして代表的なものに TinySVM[9]と YamCha[10]を組み合わせたものがある。TinySVM は SVM による識

[9] http://chasen.org/~taku/software/TinySVM/
[10] http://chasen.org/~taku/software/yamcha/

別空間を構成するツールである。与えられた各要素に該当する特徴量とクラスから 2 値表現によるベクトルを作成し、2 値分類のための境界超平面を計算している。しかし、TinySVM だけでは自然言語を処理することができない。その問題を解決するために YamCha を使用する。YamCha は自然言語のデータから、TinySVM が処理可能な 2 値のデータに変換し、また TinySVM の出力を自然言語に変換している。つまり YamCha はユーザと TinySVM を仲介する役割を担っていることになる。よって、ユーザが SVM を実現するためには、YamCha が処理できるデータ構造を入力として作成すればよい。

「太郎は花子に本を貸した」という文を例として、YamCha が処理可能なデータ構造を図 17 に示す。

特徴 1	特徴 2	特徴 3	深層格
太郎	名詞	0	動作主格
は	助詞	0	null
花子	名詞	0	対象格
に	助詞	0	null
本	名詞	0	目的格
を	助詞	0	null
貸し	動詞	1	null
た	助詞	0	null

図 17: YamCha のデータ構造

各単語につき 1 行のデータを保持し、あらかじめ決められた特徴量に対してその単語が有する情報を同じ行に記述していく。図 17 の場合、特徴 1 として単語の表記、特徴 2 としてその単語の品詞、特徴 3 では文の述語である単語に 1、それ以外の単語に 0 が記述されており、行の最後にはその単語が取り得る深層格が付与されている。このように最初の列には単語そのものが書かれることが多く、この単語の表層情報も特徴ベクトルを構成する重要な特徴量となっている。なお、文と文の間には空行を一行入れる。

作成されたファイルを、最初の行から最後の行までを一行ずつ分類器に入力する。デフォルトでは行末の文字列を深層格タグとみなし、それが正解になるように特徴量から識別器を構成していく。この時、推測のために考慮するべき特徴量の範囲を図 18 に示す。

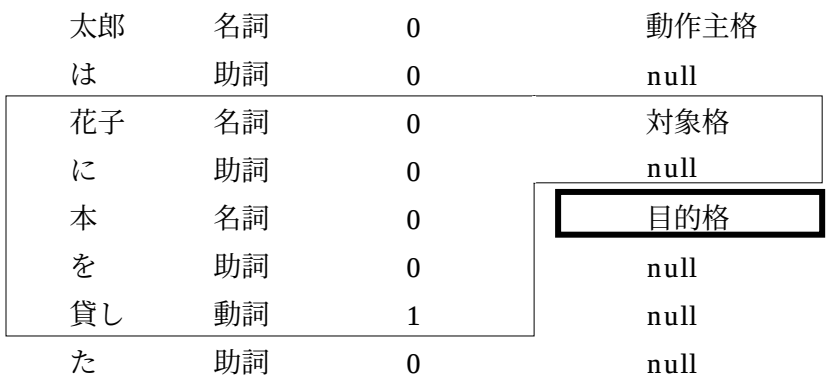


図 18: 学習時に考慮する特徴量の範囲

深層格を分類する単語が「本」の行にあるとき、この単語付与されるべき深層格である「目的格」を学習するために考慮する特徴の範囲を実線で囲ってある。第 1 列から第 3 列までの部分を静的特徴といい、前後 2 単語ずつの特徴量を考慮していることがわかる。行末の文字列はシステムによって動的に割り当てられていくタグなので動的特徴といい、図 18 の例では分類の対象となる単語の前 2 文字分を深層格分類のために考慮していることがわかる。静的特徴と動的特徴の範囲は、学習の際に指定することができる。静的特徴の場合、各特徴において考慮する範囲を個別に設定することが可能である。学習を行う際、YamCha では make コマンドを使用するが、このときに「FEATURE」というパラメータを変更することによってこの範囲を設定できる。デフォルトでは「FEATURE=”F:-2..2:0.. T:-2..-1”」となっている。F は静的特徴、T は動的特徴の範囲を示している。非連続な範囲も指定することも可能である。

このように学習の際には、YamCha ではいくつかのオプションを加えたり、パラメータを変更したりできる。他にも多値分類法の手法としてデフォルトでは Pairwise が選択されているが、OVA 選択する場合には「MULTI_CLASS」のパラメータを 2 にセットすればよい。学習の際に設定できるその他のパラメータ、およびテストの際に使用する YamCha コマンドで用いるオプションの紹介は付録 6 を参照されたい。

TinySVM と YamCha で構成されるシステムは、自然言語処理を始めとしたパターン識別分野で広く使用されている。

第 3 章　提案

これまで述べた現状を踏まえ、本研究では SVM を用いた日本語文に対する浅い意味解析の手法を提案する。

浅い意味解析の精度を上げるためにはさまざまな方法が考えられるが、本研究では 2 つの点に注目した。第 1 に、文から抽出する特徴量の追加および変更である。本研究のシステムにより出力される結果は、主たる処理を行う TinySVM と YamCha に大きく依存する。そのため分類を行うモデルを構築する材料そのものである特徴量をコーパスからどの様な規則を用いて抽出するかが精度に大きく影響する。よって深層格をより効果的に分類するための優れた特徴量を考える必要がある。第 2 には、ある単語の深層格を分類するために特徴量の範囲をどのように設定するかでかる。動的特徴の範囲と静的特徴の範囲の変更することで精度向上を図る。以上の点における 2 つの提案手法を示す。

3.1　付与する深層格の決定

まず研究の目的を達成させるために設定する深層格を決定する。本研究では、深層格として Fillmore の深層格システム（表 1）を参考にした。Fillmore による深層格システムは言語普遍の性質をもつものとして提唱されており、それは本研究の目的と合致するからである。Fillmore の深層格システムに該当する EDR 日本語コーパスの概念関係子を表 5 に示す。ただし Fillmore の「経験者格」は該当する概念関係子になかったために除外した。

表 5: 本研究で付与する深層格

概念関係子	説明	対応する Fillmore の深層格
agent	動作を起こす主体	動作主格
object	動作・変化の影響を受ける対象	道具格
implement	動作における道具・手段	対象格
source	事象の主体または対象の最初の位置	源泉格
goal	事象の主体または対象の最後の位置	目標格
place	事象の成立する場所	場所格
time	事象の起こる時間	目的格

本研究ではこれら 7 種類のタグを深層格として割り振ることにした。

3.2 提案手法 1: 文から抽出する特徴量

3.2.1 日本語コーパスの解析

有効性のある特徴量を考える目的で EDR 日本語コーパスを解析した。深層格の多くは名詞であることから、名詞と深層格の関係に関して調べた。また文中に存在する複数の深層格がどのような位置関係で共起しているかを解析した。

3.2.1.1 名詞と深層格の関係

主要な述語に対する深層格の数が 2 つ以上ある文を EDR 日本語コーパスから抽出し、各深層格が付与された単語に占める名詞の割合を調べ、表 6 に示した。

表 6: 深層格に占める名詞の割合

	総数	名詞数	割合
agent	29812	28387	95.2%
object	75736	64337	84.9%
implement	7243	5495	75.9%
source	3379	3029	89.6%
goal	22255	17218	77.4%
place	12777	12233	95.7%
time	18528	13134	70.9%
深層格	169730	143833	84.7%

各深層格が付与された単語に占める名詞の割合は少なくとも 70%を超え、さらに深層格が付与されたすべての単語 169,730 個のうち 84.7%が名詞であることがわかった。よって、ある単語に深層格が付与されるかどうかを判断する際には、名詞を限定する情報が重要となり、その情報はシステムの精度にも大きな影響を与えること推測できる。

名詞の中から深層格が付与される単語を抽出する方法として、助詞に注目する。助詞は体言の後に接続することが多い。また、助詞の機能の違いにより直前の体言が持つ意味情報も推測するができる。ここで深層格が付与された単語の直後に助詞が接続する数を各深層格について調べたものを表 7 に示す。

表 7: 深層格の次の単語が助詞の割合

	総数	次単語が助詞	割合
agent	29812	24521	82.3%
object	75736	61533	81.2%
implement	7243	5277	72.9%
source	3379	2888	85.5%
goal	22255	16357	73.5%
place	12777	11696	91.5%
time	18528	5425	29.3%
深層格総数	169730	127697	75.2%

「time」を除いて、深層格が付与された単語の直後には 70%以上の確率で助詞が来ることがわかる。これより文中で重要な意味を持つ名詞は、多くの場合その直後に助詞が共起していることがわかった。よって、助詞をその働きにより分類することで、直前の体言の特徴を表現できることが期待できる。

3.2.1.2 同一文に存在する深層格が付与された単語対の位置関係

言語類型論に基づく日本語の基本語順は「主語・目的語・動詞」[11]である。つまり、日本語では「agent」という深層格は文頭の単語に対して現れ、「object」は動詞の前の単語に現れることが多いと言うことができる。これより、日本語における深層格は現れる順番や、先頭や文末などに出現することが多いなどの位置的な規則性があることが推測される。ここで、文を 5 つに等分割して文頭から領域１、領域２とし、それぞれの領域に各深層格が付与された単語がいくつあるかを調べた。結果を表 8 に示す。

表 8: 文を 5 分割した領域内にある各深層格の数

	agent	goal	implement	object	place	source	time
領域 1	18296	1004	1085	6162	3909	632	8551
領域 2	7202	1625	1519	6257	3161	783	7334
領域 3	6103	3248	2257	9687	3477	1206	5251
領域 4	5009	8551	3128	21857	3973	1546	4364
領域 5	2185	11803	1746	26717	2323	753	1423

表 8 で顕著な特徴が見られるのが「goal」と「object」である。「goal」はおよそ 75%が文

[11] http://ja.wikipedia.org/wiki/%E8%AA%9E%E9%A0%86

の後半である領域 4,5 に集中し、「object」もおよそ 70%が後半に集中している。このように、文中における位置情報によって、各深層格の付与されやすさが推測できることがわかる。SVM はデフォルトで前後 2 単語分の特徴量しか考慮していないため、領域番号を用いた位置情報を単語に付与することは深層格の推測にも効果が得られると考えられる。

さらに文中ですべての組合せの深層格ペアを作り、それら二つの深層格がどの領域で共起しているかを調べた。結果は、付録を参照されたい。

3.2.2 提案する特徴量

本研究で EDR 日本語コーパスから抽出して使用する特徴量を提案し、その特徴量について説明する。

1. 単語

単語とは、表層的な単語そのものをベクトル化した特徴量である。

2. 品詞

品詞は、該当する単語の品詞情報のことである。EDR 日本語コーパスでは名詞、動詞、形容詞、形容動詞、副詞、連体詞、接続詞、接頭語、接尾語、語尾、助詞、助動詞、感動詞、記号、数字の 15 種類に分類されている。

3. 述語（predicate）

文中の主なる述語である単語に対して 1 を、それ以外の単語すべてに 0 を付与する 2 値の特徴量とした。EDR に記述されている意味解析結果から抽出した。EDR に付与されている意味解析結果の最初に記述されている「main」となる単語を文中の主なる述語として扱った。

4. 述語からの距離[9]

各単語が主たる述語から何単語分離れているかを整数で示したもの。述語である単語を基準として 0 とし、文頭に向かうごとに-1、文末に向かうごとに+1 が加算された整数を特徴量とした。

5. 助詞の細分類

ある単語が助詞とともに句を形成するとき、それを特別に表層格とよぶ場合がある。例えば名詞「私」と助詞「が」で形成される表層格は特にガ格といわれ、文の意味的な構成に深く関与する深層格の動作主格とすることが多い。このように、日本語の深層格には、助詞は大きく関与している。助詞はその役割によって大きく 6 種類に分けることができる。種類とその働きを表 9 に示す。

表 9: 助詞の細分類[11]

分類	働き	例
格助詞	体言につき、文の中での意味的な関係を表す	が，を，に，…
並立助詞	対等な関係に立つ語を接続する	と，や，やら，…
準体言助詞	「こと」や「もの」に置き換えられ、名詞と同じ働きをする	の
接続助詞	文と文の意味関係を接続する	が，ので，ても，…
副助詞	体言や副詞に付き全体として副詞的な働きをする	は，も，こそ，…
終助詞	文や句の末尾について特定の意味を付け加える	かな，ぞ，とも，…

表 9 から、助詞はそれぞれの役割によって付近の語の品詞や文の構造的な情報を内包していることがわかる。助詞を働きにより分類することで、それらの情報を特徴量にできると推測する。特に格助詞は先の例にも挙げたように文の意味的な関係に深く関わる場合が多い。よってこれらを明示することで、深層格の付与にも役立つと考えられる。

ここで問題なのは、表層的には同じでも複数の異なった働きをする可能性がある単語の扱いである。例えば、「が」という助詞は、直前に名詞「私」などの体言がきた場合は格助詞に分類されるが、句と句をつなぐ接続助詞にも「が」という助詞が存在する。この 2 つの「が」は異なる働きであるが、表層上は全く同じである。この問題への対応として、異なる働きをするが表層上同じである助詞を、新しい分類先として定義することにした。つまり「が」という助詞があった場合、これは「格助詞と接続助詞になりうる助詞」を表すクラスを新しく定義することにした。このようにして EDR 日本語コーパス内に存在するすべての助詞を分類したものが、表 10 である。

表 10: 助詞の細分類表

タグ	働き	例
1	格助詞（格）	を,に,へ,より,をば
2	並立助詞（並立）	やら,たり
3	準体言助詞（準体言）	（該当なし）
4	接続助詞（接続）	し,て,ば,つつ,ては,ても,ので,…
5	副助詞（副）	は,も,こそ,さえ,しか,だけ,のみ,…
6	終助詞（終）	かい,かしら,さ,せ,ぞ,って,とも,…
7	格 or 接続	が,で,から,とて
8	格 or 副	まで
9	格 or 接続 or 副	して
10	格 or 準体言 or 終	の
11	格 or 並立 or 接続 or 終	と
12	並立 or 終	や
13	並立 or 副	だの,とか
14	並立 or 接続	なり
15	並立 or 副 or 終	か
16	接続 or 副	でも,なら
17	接続 or 終	けど,だって
18	副 or 終	ったら,ってば,がな,い
19	助詞による連語	では,じゃ,じゃあ,もって…
0	助詞以外	―

助詞の細分類による特徴量では、表 10 の通り 20 種類（「3」を除くと 19 種類）に分類したときの番号を各単語の特徴量としている。なお、助詞以外の品詞のものには「助詞以外」を表すタグをつけている。

6. 助動詞の細分類

助詞と同様に、助動詞についても機能に基づいて分類し、その分類番号を特徴量として付与した。助動詞は動詞と形容動詞の後に接続できるが、特に動詞に接続する際にはその動詞が意味的に影響のある範囲内における意味的構造を限定する働きを持っていると考える。例えば動詞「逃げる」に受身の助動詞「らる」が接続した「逃げられる」という句を考えたとき、これを含む文は「誰が」「誰に」逃げられたのかが表現されていることを推測することができる。これに対し、受身の助動詞がない「逃げる」という動詞を含む文のときには、「誰が」「どこから」「どこへ」といった表現がされていることを推測できる。このように、助動詞の意味によって、その助動詞が接続する動詞がどのような意味を表現して

いるかを限定し、それによって考慮すべき深層格が限定されるのではないかと考えた。

助動詞の細分類は、コーパスからすべての助動詞を抽出してから意味ごとに分類した。コーパスの中で助動詞とされているものは計 190 語あったが、辞書[12] [13]を使用して調べ、実際に助動詞と認められたのは 73 語であったので、これらを分類した。分類は「助動詞ではない」というクラスも含め、合計 36 クラスに分類した。分類の基準を表 11 に示す。

表 11: 助動詞の意味による分類

class	意味	例	class	意味	例
1	意識的動作の完了	つ,て	19	直接体験過去	き,し
2	依頼･許可･敬意	せる,せれ	20	丁寧･敬意	ましょ,ます
3	打消し	じ	21	丁寧･断定･強調	でしょ,です
4	打消し･勧誘	ない,なかろ,なく,なけれ	22	比況	ごとく
5	打消し推量･打消し意思･禁止･勧誘	まい	23	様態	そうだ,そうで,そうな
6	受身･尊敬･可能･自発	られ,られる,れ,れる,れれ	24	推量･意志･義務･可能	べき,べく,
7	打消し･禁止･義務･許可･依頼	ず,ん	25	打消推量･禁止	まじ
8	過去･完了･確認･命令･決意	た,たろ	26	反実仮想	ませ
9	間接体験の過去	けり	27	2,12,20 参照	せ
10	願望	たい,たがら,たがる	28	8,11,16 参照	たら
11	完了･存続	ら,り	29	11,16	たり,たる,たれ
12	使役･尊敬	さす,させ,しめ,しめる	30	17,18	なら
13	推定	らし	31	18,24	なり,なる,なれ
14	推量	ろう	32	7,27	ぬ,ね
15	推量･意思･当然･婉曲	う,め	33	22,31	まし
16	断定･強調	だ,だっ,だろ,で,な	34	6,11	る,れよ
17	断定･存在	に	35	21,26	ざる
18	断定･認定･尊敬	じゃ,じゃろ	0	助動詞でない	-

[12] 「大辞泉」，小学館
[13] 「大辞林」，三省堂

述語動詞が取り得る深層格を限定することを期待して、これらの分類を特徴量として付与した。

7. 助動詞の活用による細分類

助動詞の意味による分類に続き、助動詞の活用形による分類も行った。活用形はその後に続く単語の性質を限定することができる。例えば連体形であればその後に体言が続き、連用形であれば用言が続くことがわかる。しかし助詞と同様に、助動詞も表層的な情報だけでは活用形を一つに限定することができない。よって、複数の活用形になりうる語は新しいクラスとして分類を行った。分類基準を表 12 に示す。

表 12: 助動詞の活用による細分類

クラス	活用形	例
1	已然	すれ,せれ,なけれ,め,れれ
2	已然･命令	だ,なれ,れよ
3	終止	う,き,けり,さす,じゃ,ず,etc
4	終止･連体	しめる,せる,たい,etc
5	終止･連体･已然	じ,らし,ん
6	未然	じゃろ,たがる,だっ,だ,etc
7	未然･已然	たる,なら
8	未然･連用	ごとく,させ,しめ,せ,て,etc
9	未然･連用･已然･命令	ね
10	命令	ませ
11	連体	ざる,し,する,たろ,な,etc
12	連用	そうで,たり,なく,に
13	連用･終止	たれ,なり,り
14	連用･終止･連体	まし
0	助動詞ではない	-

8. 文中における領域情報

文を 5 つの領域に分割して文頭から領域 1,領域 2 と順に称し、各単語が所属する領域番号を特徴量として付与した。分割は文の単語数を基準としている。文の単語数が 5 の倍数であれば(単語数)/5 の算出結果が各領域の単語数となる。文の単語数が 5 の倍数でない場合は、余りの数を領域 1 から順番に 1 ずつ分配していく。

これら 8 つの特徴量を用いて実験を行い、どの特徴量が日本語の浅い意味解析に有効であ

るかを考察する。これらの特徴量のうち、特に 5 から 8 が本研究特有のものである。

3.3 提案手法 2: 特徴量の範囲

本研究では特に静的特徴の範囲と動的特徴の範囲に注目する。静的特徴は深層格を分類する単語の前後何語ずつの情報を考慮するか、その範囲を広げたり縮めたりすることで精度にどのように影響を与えるかを調べる。これにより、処理の対象となる単語がその付近の単語の意味的役割にどれだけの影響を与えているか、また付近の単語がその単語の意味的役割にどれだけの影響を与えているかを解明することが出来ると考える。また動的特徴の範囲は、システムが付与してきたその文におけるすべての深層格を処理の対象としてみる。そうすることで、同じ深層格が同一の文で何度も付与されたり、他の深層格との共起関係も SVM が学習したりすることを期待する。

3.4 システムの構成

PWHMJ2004 [5][6]と同様、SVM の実現には YamCha と TinySVM を使用する。これらのツールを用い、EDR 日本語コーパスを対象とした浅い意味解析を行う。提案するシステムの概要を図 19 に示す。

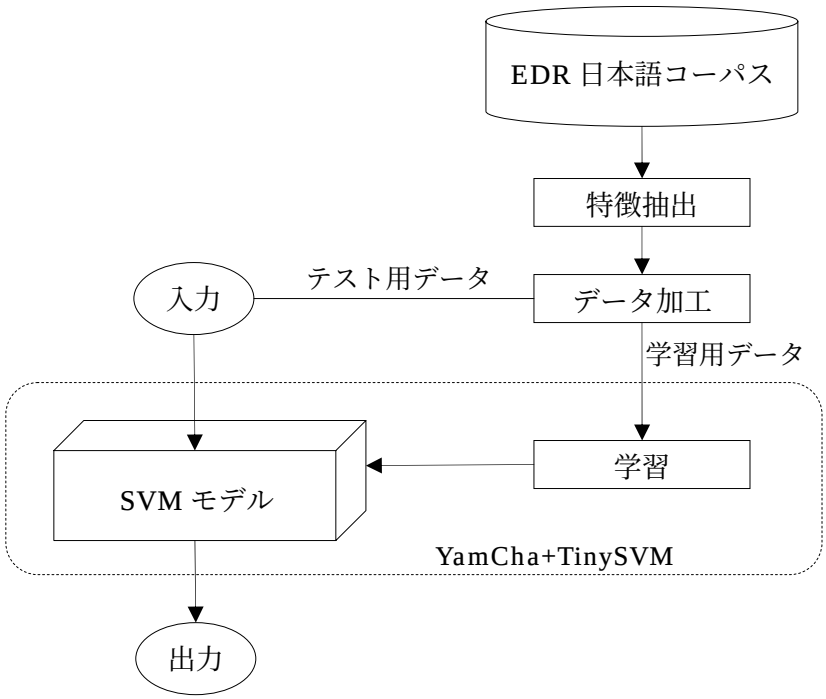


図 19: システム概要

まずEDR日本語コーパスの文から提案手法1で述べたSVMの特徴量を抽出するとともに、YamCha が処理可能なデータ構造に変換する。その後、文中に 2 つ以上の深層格を持つ文のみを抽出する。先に述べたように EDR 日本語コーパスには各解析における判断基準の不均等が存在し、この問題をある程度解消させるために本研究では処理対象とする文を絞り込んだ。処理対象を洗練させることで評価となる指標を高めることも可能だが、本研究ではそのような目的で文の絞込みは行わない。さらに 10-fold cross validation 用のデータセットを作成するためのデータ加工を行う。10-fold cross validation とは、全データを 10 分割したうちの 9 つで学習を行って SVM モデルを構築し、残りの 1 つから深層格を除いて未知データとみなして SVM モデルに入力するという処理を計 10 回行う実験手法のことである。図の点線で囲われた個所は、YamCha と TinySVM で処理を行う。

3.5 評価指標

実験結果の評価指標として適合率、網羅率、F_1値の 3 つを用いた。算出の仕方を以下に示す。

- 適合率$(precision) = \dfrac{\text{システムが付与し、正 しかった深層格の数}}{\text{システムが付与した深 層格の総数}}$
- 網羅率$(recall) = \dfrac{\text{システムが付与し、正 しかった深層格の数}}{\text{コーパスにある深層格 の総数}}$
- $F_1 = \dfrac{2 * precision * recall}{precision + recall}$

実験は 10-fold cross validation で行う。F_1値は、適合率と網羅率を用いて算出されるため、F_1値を総合的な評価指標とする。

3.6 パラメータの決定

実験を行う前に設定しておく必要があるパラメータを決定するために予備的な実験を行った。実験結果を考察することで多値分類法を選択し、学習に用いる文例数を選択する。それ以降の実験では選択されたパラメータ設定において学習をする。

3.6.1 多値分類法の決定

多値分類法の決定は学習で構築するモデルに影響を与える。深層格付与に適したモデル

を構築するために設定する必要のある重要な項目の一つである。多値分類法は Pairwise 法と One-versus-All 法の 2 種類から選択する。

3.6.1.1 実験概要

実験は FrameNet コーパスを用いて行った。データは Clothing フレームから 3049 文、Containers フレームから 1984 文、Custom フレーム 247 文、Food フレーム 1591 文、Motion_direction フレーム 161 文、Natural_features フレーム 1819 文、Observable_bodyparts フレーム 2506 文、Quantity フレーム 746 文、Self_motion フレーム 6206 文、Social_event フレーム 497 文、Social_interaction_evaluation フレーム 1007 文、Subject_stimulus フレーム 3917 文の、計 13 フレームにおける文例を用いた。それぞれのフレームにおいて frame element を SVM で付与する処理を、Pairwise 法と One-versus-All 法の双方で行い、その適合率、網羅率および F_1 を比較した。

3.6.1.2 結果

フレームごとに算出された Pairwise 法（PW）、One-versus-All（OVA）法双方における適合率、網羅率、F_1 を表 13 に示す。

表 13: 予備実験結果

フレーム名	文例数	適合率		網羅率		F_1	
		PW	OVA	PW	OVA	PW	OVA
Clothing	3049	0.819	0.826	0.777	0.788	0.793	0.806
Containers	1984	0.843	0.85	0.801	0.813	0.821	0.831
Custom	247	0.49	0.514	0.397	0.402	0.439	0.451
Food	1591	0.731	0.738	0.709	0.715	0.72	0.727
Motion_direction	161	0.535	0.595	0.44	0.499	0.483	0.543
Natural_features	1819	0.875	0.88	0.822	0.835	0.848	0.857
Observable_bodyparts	2506	0.91	0.913	0.882	0.886	0.896	0.899
Quantity	746	0.825	0.834	0.866	0.876	0.845	0.854
Self_motion	6206	0.752	0.76	0.692	0.707	0.721	0.733
Social_event	497	0.712	0.73	0.563	0.596	0.628	0.656
Social_interaction _evaluation	1007	0.689	0.655	0.573	0.622	0.626	0.638
Subject_stimulus	3917	0.793	0.794	0.712	0.715	0.75	0.752

3.6.1.3 評価・考察

表 13 の適合率、網羅率、F_1 のそれぞれの評価指標において、「Pairwise 法と One-versus-All 法の違いによる結果に差はない」という仮説の下で t 検定を行った。その結果、網羅率と F_1 の指標については、1%の危険率で統計的に有意差があることがわかった。よって、Pairwise 法と One-versus-All 法に差がないとは言えず、仮説は棄却される。適合率に関しては、仮説を必ずしも否定できないとなったが、システムの総合的な評価指標である F_1 において有意差がないとは言えない。表 13 をみると、Pairwise 法より One-versus-All 法の結果の方が高い精度であることがわかるので、本研究では One-versus-All 法を多値分類法として採用することにする。

3.6.2 文例数の決定

学習の際に用いる文例数は、学習データに含まれない単語がテストの際に未知語となるためにある程度の規模が必要となるが、規模が大きすぎるとノイズが多く含んだり学習に莫大な時間がかかったりしてしまう。よって適切な文例数を設定する必要がある。

3.6.2.1 実験概要

文例数は 1,000 文、2,000 文、3,000 文、4,000 文、5,000 文、10,000 文、15,000 文、20,000 文における 8 つのパターンで実験を行った。このときの特徴量は提案した単語、品詞、述語、述語からの距離、助詞の細分類の 5 つを用いた。

3.6.2.2 結果

結果は図 20 の通りである。

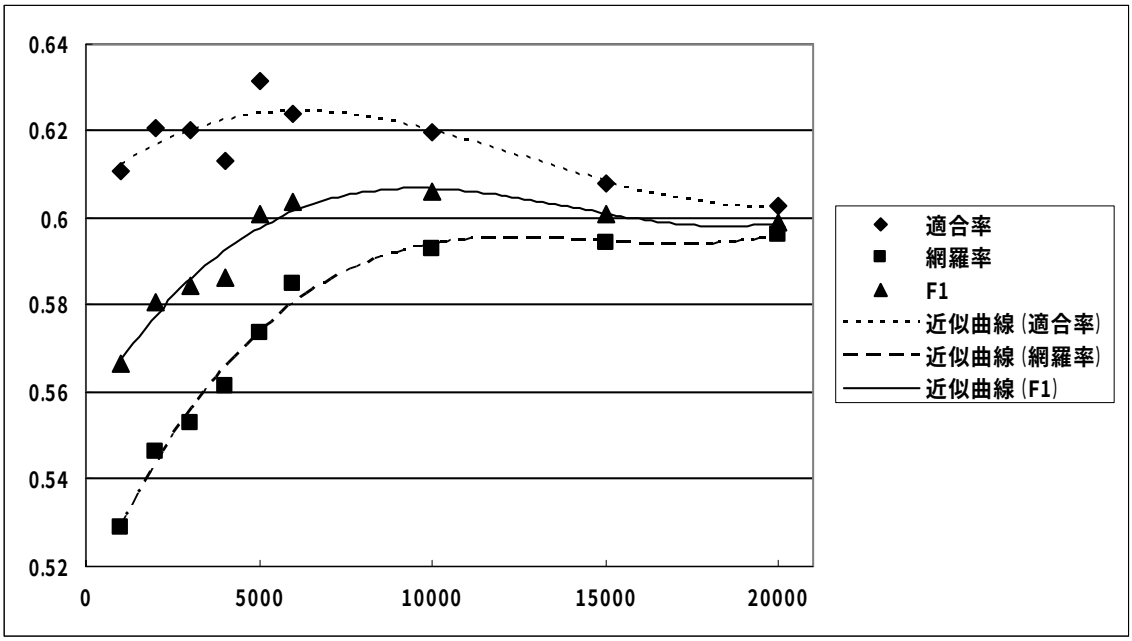


図 20: 文例数変化における精度

図 20 における曲線はそれぞれの評価指標における近似曲線である。適合率、網羅率ともにその経過は 2 次の多項式における表現が期待できるが、文例数が 20,000 文の結果は、15,000 文の結果よりも必ずしも劣っているとはいえないため、近似曲線は 3 次の回帰曲線で表現した。

3.6.2.3 評価・考察

図 20 における適合率の近似曲線を見ると文例数が 5,000 文から 10,000 文の間にピークが現れており、網羅率の近似曲線は 10,000 文まで増加を続け、その後は安定している。さらにシステムを総合的に評価する指標である F_1 値の近似曲線のピークは 10,000 文付近に現れていることが分かる。よって、0 から 20,000 文までの文例数で学習を行う際、理論上は 10,000 文付近で学習することがよいということがわかる。

ここで、文例数とその学習にかかった時間の関係を図 21 に示す。実験に用いた計算機の性能は、CPU が Pentium4 3.4Ghz、二次キャッシュ 2MB、メモリ 2GB である。

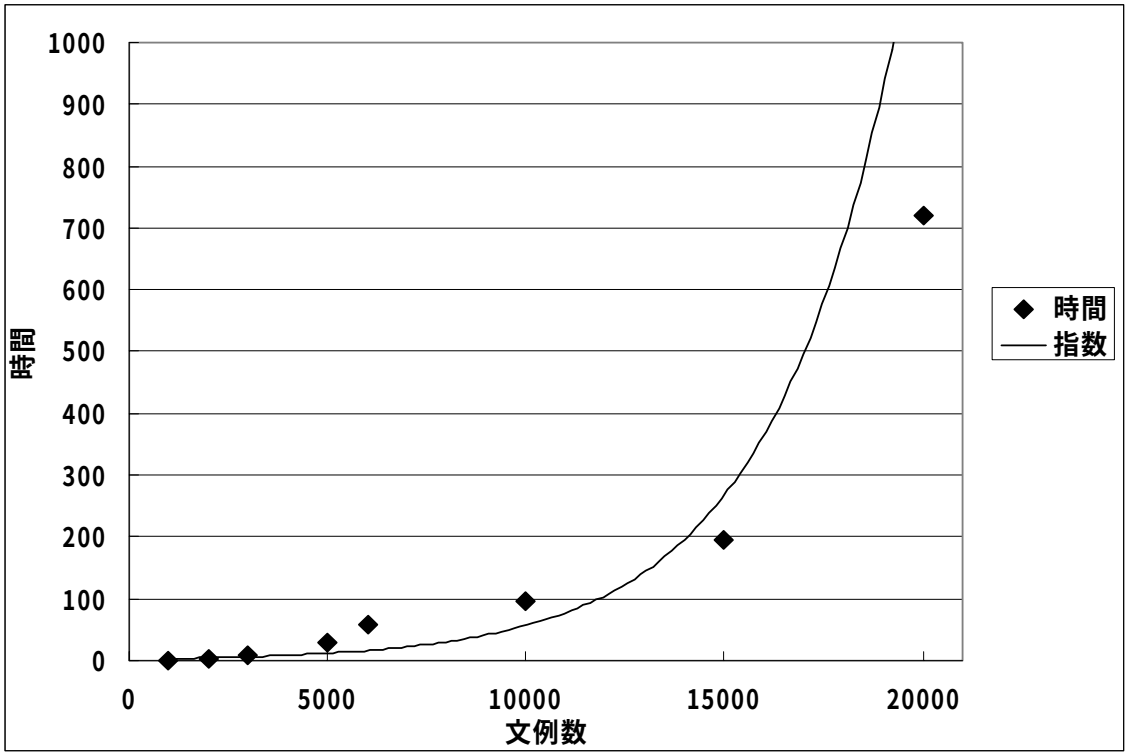


図 21: 文例数の学習時間の関係

学習時間は文例数が 5,000 文を超えたあたりから指数関数的に増加をしている。5,000 文で約 30 時間なのに対し、10,000 文の学習にはおよそ 4 日間かかった。本実験ではシステムの精度と学習時間を考慮し、5,000 文で行うことにする。

第 4 章　実験・結果

本実験では予備的に行った実験をもとに One-versus-All 法を多値分類法として採用、文例数を 5,000 文とした上で、EDR 日本語コーパスを用いた 7 種類の深層格を付与する実験を行う。まず提案手法 1 で示した**特徴量**を追加、変更することによる実験を行い、次に提案手法 2 で示した静的特徴、動的特徴の範囲を変化させて実験を行った。

本実験ではコーパスの絞り込みのため、文中の述語動詞との関わりを示す深層格が 1 文で 2 つ以上存在する文のみを抽出した上で、YamCha への入力ファイルを作成した。これは EDR 日本語コーパスの各解析における判断基準の不均等を排除するためではなく、推測する深層格が少ない文を扱う事によって深層格付与に消極的なシステムを構築しないためである。さらに 1 文あたりの単語数は平均で 23.8 単語あるが、これらの中に深層格が付与されるものが 1 つしかない場合、仮に正しく深層格を付与したとしても言語普遍の**特徴量**が得られたとはいいがたい。よって 2 つ以上の単語が述語動詞と関わる文を処理対象とすることにした。

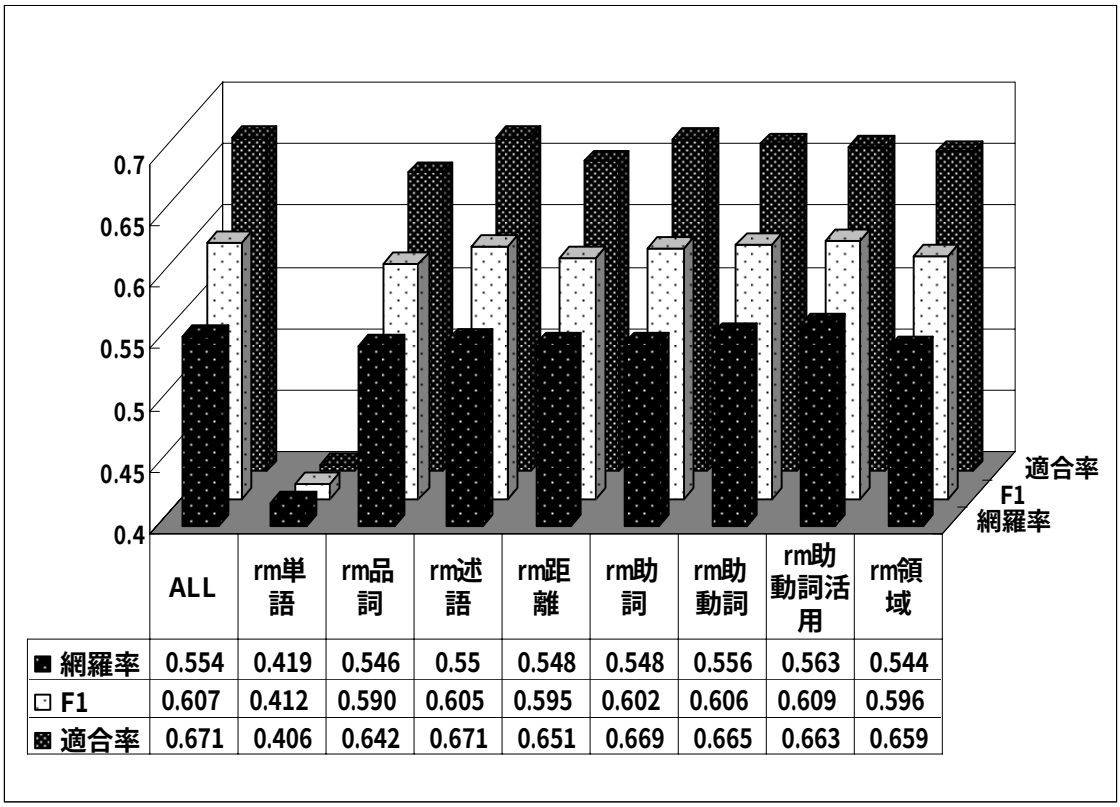
4.1　実験 1：文から抽出する特徴量の有効性

4.1.1　実験の目的

提案した 8 つの**特徴量**がそれぞれの精度にどのように影響を与えているかを調べることを目的として実験を行う。その結果から、どのような**特徴量**が日本語文への浅い意味解析にとって有効であるかについても考察する。

4.1.2　実験・結果

まず提案した 8 つすべての特徴量を用いて実験を行った。さらに各特徴量がどれだけ精度に影響を与えるかを調べるため、8 つの特徴量から 1 つを除外した 7 つの特徴量でそれぞれ実験を行った。結果を図 22 に示す。

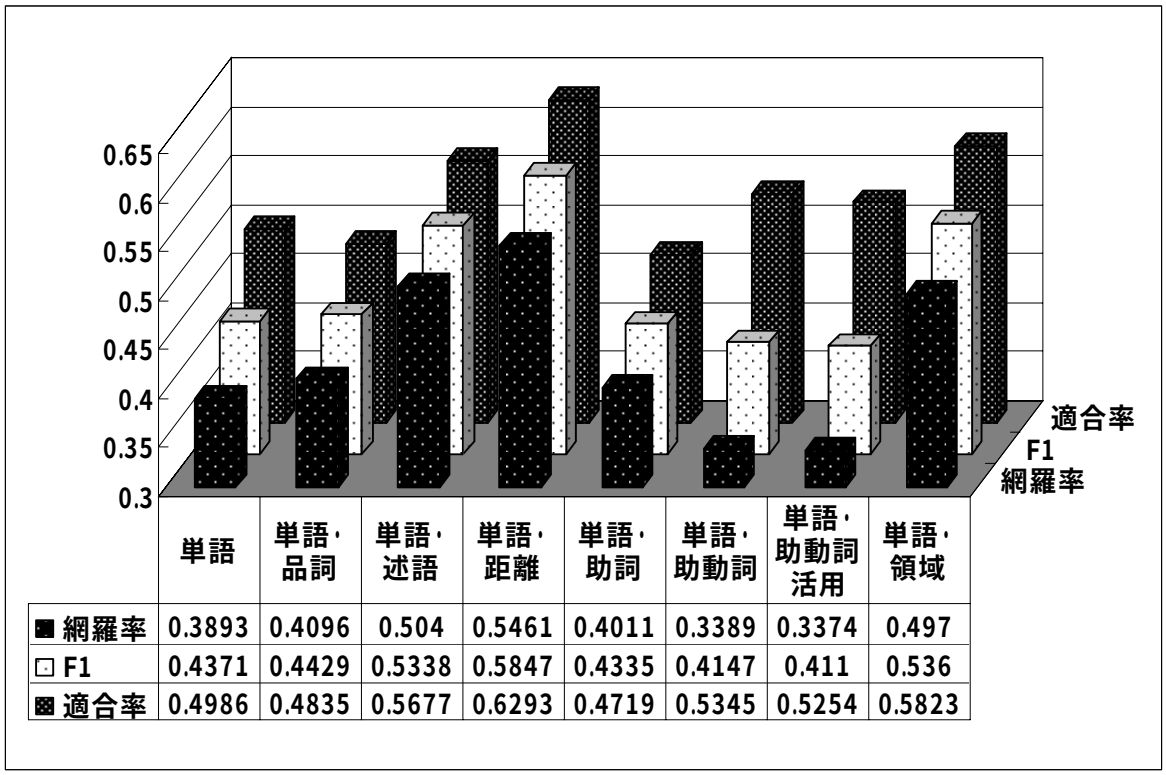


	ALL	rm単語	rm品詞	rm述語	rm距離	rm助詞	rm助動詞	rm助動詞活用	rm領域
網羅率	0.554	0.419	0.546	0.55	0.548	0.548	0.556	0.563	0.544
F1	0.607	0.412	0.590	0.605	0.595	0.602	0.606	0.609	0.596
適合率	0.671	0.406	0.642	0.671	0.651	0.669	0.665	0.663	0.659

図 22: 8 つの特徴量を用いた実験結果

図 22 において、「All」が 8 つの特徴量をすべて用いた結果であり、「rm」がついた項目はその特徴量を抜いた他 7 つの特徴量での結果である。

図 22 を見ると「単語」を除いたときの精度の低下が顕著なので、これが最も重要な特徴量であることが容易にわかる。他のどの特徴量が有効であるかを調べるために、「単語」と残りの 7 つの特徴量のうちの 1 つを組み合わせた計 2 つの特徴量で、どの程度の精度を得ることができるかを調べた。結果は図 23 に示す



	単語	単語，品詞	単語，述語	単語，距離	単語，助詞	単語，助動詞	単語，助動詞活用	単語，領域
網羅率	0.3893	0.4096	0.504	0.5461	0.4011	0.3389	0.3374	0.497
F1	0.4371	0.4429	0.5338	0.5847	0.4335	0.4147	0.411	0.536
適合率	0.4986	0.4835	0.5677	0.6293	0.4719	0.5345	0.5254	0.5823

図 23: 単語ともう 1 種類の特徴量を用いての実験結果

一番左が単語だけでの実験結果、右にあるのが単語ともう 1 種類の特徴量を加えて実験した結果である。

4.1.3 評価・考察

本実験において、最も高い F_1 値だったのは 8 つの特徴量から助動詞の活用による細分類を除いて学習した 60.9%であった。図 23 を見ても、単語のみで学習するよりも助動詞の活用による細分類を含めたときに精度が低くなるため、この特徴量は精度を悪化させる特徴量であることがわかる。助動詞の意味による分類も、精度が低下することはなかったが、必ずしも深層格付与に有効であるとは言えない。その原因として、EDR 日本語コーパスにおける形態素解析の質が挙げられる。今回、助詞の細分類、助動詞の細分類、助動詞の活用による細分類の 3 つの特徴量は、EDR 日本語コーパスから取り出したすべての助詞、助動詞を規則に従い細分類している。しかし形態素解析の結果として助詞および助動詞とつけられた単語でも、語彙上そうとは認められない単語が多数存在した。よって品詞の特徴量では「助詞」や「助動詞」とされていても、助詞の細分類、助動詞の細分類、助動詞の活用による細分類では「助詞ではない」や「助動詞ではない」ということを表している特徴量となってしまうことで、データの質を落としてしまったと考えられる。助動詞でも同

様のことが言える。このような EDR 日本語コーパスの形態素解析の質が原因の 1 つであると考えられる。

また、特徴量としての単語を除いた 7 つの特徴量の中で精度の改善に効果的なのは述語からの距離であることがわかる。これら 2 つの特徴量で実験した結果である F_1 値 58.5%は、本実験の最高精度に肉薄している。述語からの距離情報は Shibui[9]の研究により英語文への深層格付与に有効であるとわかったが、日本語文においても同様の結果であることがわかった。領域情報もまた位置情報の一種であり、図 22 と図 23 からも有効性が示されている。分割数が 5 であったが、より細かい分割を行うことで特徴量の情報量を高めることができ、より有効な特徴量となると考える。

2 つの特徴量のみで F_1 値を 58.5%まで高めることができたが、8 つすべての特徴量を用いても F_1 値は 60.7%で上げ止まる。品詞を細分類した特徴量は効果が見られず、単語の特徴量による表層的な情報のみで十分であることがわかり、述語からの距離や文中における領域情報といった位置情報が有効であるという結果となった。

4.1.4 提案手法 1 の結論

本実験では述語からの距離と領域情報が有効な特徴量であり、これから SVM を用いた日本語文への深層格付与には、位置情報を表現した特徴量が有効であると言える。また、単語の表層的な情報は非常に多くの情報を内包しているため、助詞や助動詞などの機能語を細分類することによるメリットはない。

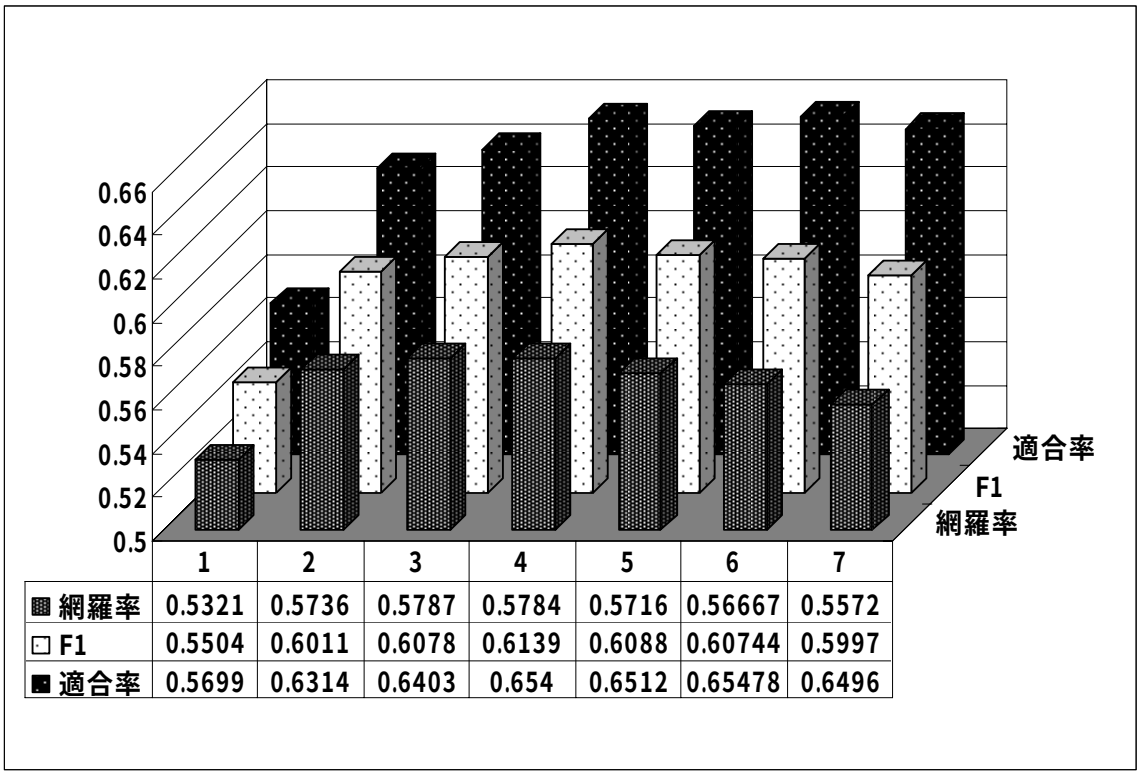
4.2 実験 2：特徴量の範囲の変化による精度への影響

4.2.1 実験 2 の目的

静的特徴と動的特徴の範囲を変化させることによる精度への影響を調べることを目的として実験を行う。その結果から、日本語文への浅い意味解析において SVM を使用する際に、どのような範囲で特徴量を学習していくのがよいのかを考察する。

4.2.2 結果

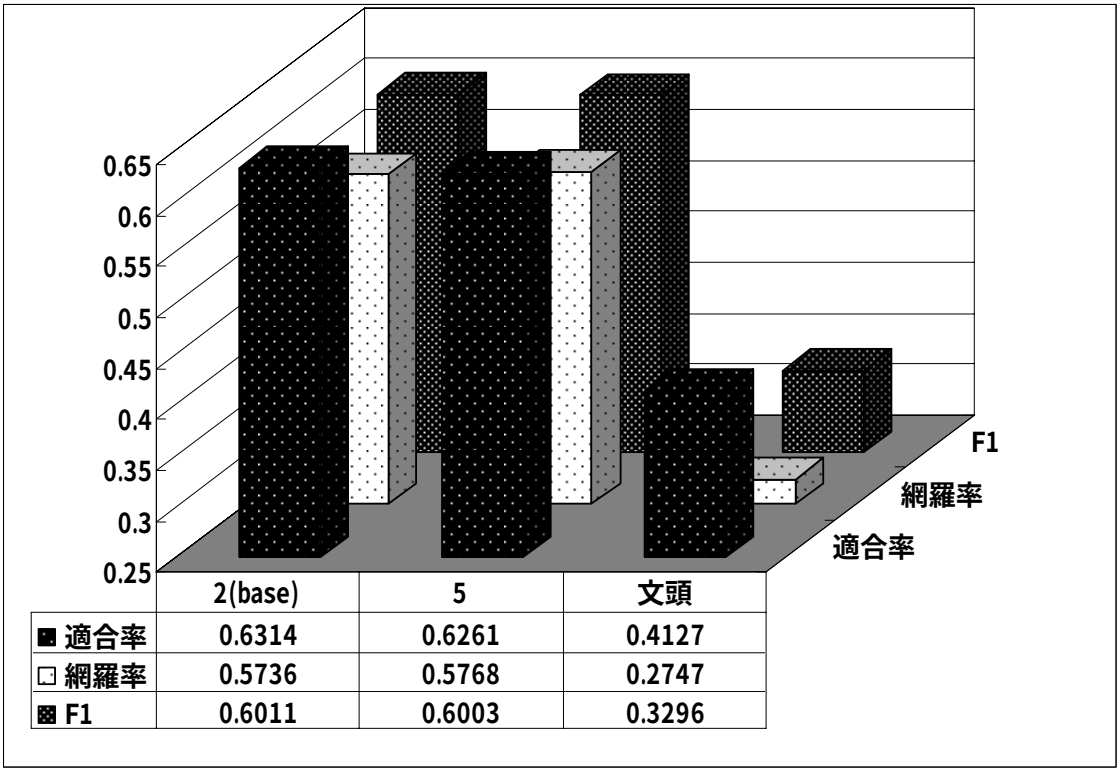
YamCha では静的特徴は深層格を分類する単語からの距離が 2 までの特徴量を標準で考慮している。この距離を 1 から 7 まで変化させてそれぞれ実験を行った。結果は図 24 に示す。



	1	2	3	4	5	6	7
網羅率	0.5321	0.5736	0.5787	0.5784	0.5716	0.56667	0.5572
F1	0.5504	0.6011	0.6078	0.6139	0.6088	0.60744	0.5997
適合率	0.5699	0.6314	0.6403	0.654	0.6512	0.65478	0.6496

図 24: 静的特徴の範囲変化における精度

また、動的特徴は深層格の分類に、1 つ前と 2 つ前の単語のものまでを考慮に入れている。これを 5 つ前まで考慮したものと、今までシステムが付与してきた動的特徴すべて、つまり文頭までを考慮したものとの 2 通りで実験を行った。結果は図 25 に示した。



	2(base)	5	文頭
適合率	0.6314	0.6261	0.4127
網羅率	0.5736	0.5768	0.2747
F1	0.6011	0.6003	0.3296

図 25: 動的特徴の範囲変更における精度

動的特徴の範囲を文頭まで設定する際には、YamCha での動的特徴のパラメータ設定を「T:-1..-100」とした。これはコーパスに含まれる文は高々2 桁の単語で構成され、さらに YamCha が必ずしも動的特徴を 100 単語前まで考慮するわけはなく、範囲の最大となる文頭までを動的特徴の範囲であると柔軟に解釈できるためである。

4.2.3 評価・考察

図 24 を見ると、静的特徴の範囲が前後 1 語ずつから 2 語ずつになったところで顕著な精度の向上が見られる。また、範囲を前後 2 語から 1 語ずつ増やしても大きな変化がないことより、対象となる単語から前後 2 語の範囲内の単語との関係に特に重要な情報が含まれていると推測できる。「太郎」という名詞の直後に「が」という助詞があった場合、「太郎」は「Agent」と推測しやすいと例証したが、このような直後の単語だけではなく、前後 2 つに存在する単語との共起情報がより重要な手がかりとなると考えることができる。そこで、深層格が付与されるべき単語の前後 2 語の単語がどのような品詞であるかを調べた。結果を表 14 に示す。

表 14: 深層格が付与される単語の前後 2 語の品詞

	2 語前	1 語前	深層格	1 語後	2 語後
名詞	61625	24775	-	57470	63102
動詞	50710	2288	-	2217	55607
形容詞	4192	38	-	670	2676
形容動詞	4138	778	-	1432	3663
副詞	1667	3469	-	3144	5534
連体詞	697	5649	-	5678	2085
接続詞	1110	634	-	5424	262
接頭語	565	2343	-	1487	1300
接尾語	7165	2497	-	7143	6383
語尾	35283	50120	-	11562	3682
助詞	34555	67578	-	133858	61406
助動詞	25163	41533	-	1782	3766
記号	8850	20224	-	8162	28666
数字	2119	5673	-	2991	6662

ここで注目すべきは、2 語後の単語の動詞の数である。深層格が付与される単語のうち、55,607 個の 2 語後には動詞が表れるという結果だが、このうち 49,520 個（89.1%）が文の主要な述語となっていることがわかった。ここで、コーパス内にある主要な述語となる動詞は 70,053 個ある。つまり、文の主要な述語となるすべての動詞の 70.7%にあたる単語の 2 語前の単語に対して深層格が付与されることがわかった。また、2 語前の動詞に関しても、33.9%が述語動詞となり、2 語後の動詞と合算すると 95.3%の述語動詞がこの範囲内に出現することになる。よって静的特徴の範囲を前後 2 語にしたことにより、述語となる単語を深層格推定の範囲内により多く含めることができていると言える。これが静的特徴の範囲を前後 2 語にすることによる精度の向上の理由であると考える。図 24 から、静的特徴の範囲を前後 4 語より広くした場合にはノイズが入ることがわかるので、前後 4 語が最適であると言える。

また図 25 を見ると、本実験では動的特徴の範囲を 2 語前までとするときが最も高い精度であった。

4.2.4 提案手法 2 の結論

述語が動詞の際にはその 2 語前に深層格を共起する場合が多く、その深層格を正確に付与するためには静的特徴の範囲を最低でも前後 2 語ずつにするのがよい。本実験では前後 4 語ずつが最適であることがわかった。

第 5 章　考察

本研究では SVM による日本語文に対する深層格付与の結果として、適合率 65.4%、網羅率 57.8%、F_1値 61.4%を得た。本章ではこの数値を他の浅い意味解析の研究と比較することで考察する。

5.1　渋木[7]との比較

渋木の研究では、処理対象の文を絞るためにコーパスにいくつかの制限を加えている。まず本研究と同じく、深層格が 1 文に 2 つ以上ある文に絞っている。また使役・受動を示す助動詞を含まないことと、機能語が字面上存在することでさらなる絞込みを行っている。さらに、深層格を付与する対象となる単語にも制限を設けている。渋木は図 13 で示したような依存関係を発見する処理を行っているが、係り先となる動詞はコーパス内に 50 回以上出現するもののみを使用し、係り元の単語は名詞でなければならない。これらを勘案して、よりコーパスへの制限、および深層格付与対象への制限を行っていない状況下で実験を行った本研究が、渋木の適合率 54.0%を上回ったことは評価できる。

しかし、渋木は付与する深層格の種類が本研究とは異なるため、渋木と同様の深層格を付与する実験を行った。用いた特徴量は単語、品詞、述語、述語からの距離、助詞の細分類の 5 つである。実験結果を表 15 に示した。

表 15: 付与対象となる深層格を渋木と同様の 11 種類にした実験結果

文例数	適合率	網羅率	F1
1000	0.6023	0.5722	0.5871
5000	0.6038	0.5967	0.6001

適合率は 60%を超え、本研究の優位性が示された。本システムは付与する深層格に依存しない分類器であるとも言える。

5.2　CoNLL:Shared Tasks での研究結果との比較

Mitsumori[8]の研究は CoNLL-2005:Shared Tasks に則ったものである。CoNLL（Conference on Computational Natural Language Learning）は ACL（Association for Computational Linguistics）[14]が主催する会議の 1 つであり、規格を定めたタスクを行う

[14] http://www.aclweb.org/

研究を集めてその精度を競う催しである。毎年そのタスクは異なるが、CoNLL-2004 と CoNLL-2005 では Semantic Role Labeling が採用された。これは、PropBank コーパスに従い、argument を判別、分類する精度を競うものである。英語文に対する浅い意味解析のタスクであるとも言える。Mitsumori は CoNLL-2005 にて、SVM を用いた意味役割タグ付けの手法を提案した。結果は、適合率 74.15%、網羅率 68.25%、F_1値 71.08%であった。また、CoNLL-2004 において SVM を使った研究である Park[12]の結果で、適合率 65.63、適合率 72.43%、F_1値 63.99%であった。

表 16: CoNLL での研究結果との比較

	適合率(%)	網羅率(%)	F_1(%)
Park	65.63	62.43	63.99
mitsumori	74.15	68.25	71.08
本研究でのシステム	65.4	57.8	61.4

EDR コーパスはすべての単語の意味構造を詳細に記述しようと数多くの概念子が存在していたが、これに対し PropBank の argument は述語との関係に特化しているために、PropBank の方がより浅い意味解析に適したコーパスであるといえる。よって、EDR コーパスを用い、日本語文に対して行った SVM による浅い意味解析の初期の研究である本研究はよい結果であると言える。

第 6 章　結論

本研究では、言語普遍とされる深層格を日本語文に対して付与する処理を、SVM を用いて行った。提案手法の評価の結果、適合率 65.4%、網羅率 57.8%、F_1値 61.4%という精度を得る分類器を構築することができた。既存の研究結果と比較しても、日本語文に対して行った SVM による浅い意味解析の初期の実験としてはよい結果が得られた。また、述語からの距離や文中における領域情報などの位置情報を表現した特徴量を用いることが有効であるとわかった。実験 2 の考察では、深層格を分類するターゲットとなる単語の前後 2 語ずつの中に文の主要な述語動詞が 95.3%の確率で出現することがわかった。日本語文への深層格付与をする場合、静的特徴の範囲を指定するパラメータは、前後 2 語ずつから 4 語ずつの間に設定しておくのがよいことがわかった。

しかし他分野へ応用するためにはより高い精度が求められる。さらに精度を上げるためには、まず領域情報の特徴量において分割数を増やすことが考えられる。また、本研究では述語と深層格が静的特徴の範囲内にあることが精度向上に寄与することがわかったが、文末の単語に付与されることが多い述語と文頭の単語に付与されることが多い深層格は、深層格分類の際に同じ静的特徴の範囲になる可能性が極めて低い。述語の付近にある深層格だけではなく、文頭に近い位置にある単語への深層格を的確に付与するための特徴量も必要である。例えば「agent」は 66.7%が文を 5 分割したうちの前半 2 領域に存在する。さらに、すべての「agent」うち、文の一番初めの単語に付与される確率を調べたところ 30.4%であり、また「agent」が文の最初に出現する名詞に付与される確率は 37.9%であった。このように、位置情報を解析することより得られた日本語文の位置的な特性を利用して、新たな特徴量をコーパスから抽出することで、精度の更なる向上につながると考える。

参考文献

[1] 荒木 健治. “自然言語のことはじめ　言葉を覚えて会話の出来るコンピュータ,” 森北出版 (2004).

[2] 田中 穂積. “自然言語　―基礎と応用―,” 電子情報処理学会 (1999).

[3] 栗田多喜夫. “サポートベクターマシン入門,” (2002).

[4] Nello Cristianini, John Shawe-Taylor, “サポートベクターマシン入門,” 共立出版, (2006).

[5] Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James H. Martin, Dan Jurafsky, “Shallow Semantic Parsing using Support Vector Machines,” TR-CSLR-2003-03, University of Colorado (2003).

[6] Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James H. Martin, Dan Jurafsky, “Shallow Semantic Parsing using Support Vector Machines,” University of Colorado (2004).

[7] 渋木英潔, 荒木健治, 栃内香次. “一文一格の原理と深層格選好に基づいた日本語深層格規則の自動獲得手法,” 情報処理学会研究報告, vol.2003, no.076, pp15-22 (2003).

[8] Tomohiro Mitsumori, Masaki Murata, Yasushi Fukuda, Kouichi Doi, Hirohumi Doi. “Semantic Role Labeling Using Support Vector Machines,” CoNLL-2005 Shared Task（2005）.

[9] Nobukazu Shibui, Akito Sakurai. “Framenet-Based Shallow Semantic Parsing with a POS Tagger,” 情報処理学会研究報告, Vol.2004, No.125, pp187-190 (2004).

[10] 伊藤 武彦, 田原 俊司, 朴 媛淑. “文の理解にはたす助詞の働き－日本語と韓国語を中心に－,” 風間書房, (1993).

[11] 林 四郎, 南 不二男, 野元 菊雄, 国松 昭. “例解新国語辞典,” 三省堂 (2001).

[12] Kyung-Mi Park, Young-Sook Hwang and Hae-Chang Rim, “Two-Phase Semantic Role Labeling based on Support Vector Machines,” CoNLL-2004 Shared Task (2004).

[13] 山田 寛康, 工藤 拓, 松本 裕治. “Support Vector Machines を用いた日本語固有表現抽出,” 情報処理学会論文誌, Vol 43, No. 1, pp.43-53.

[14] 原田 実, 田淵 和幸, 大野 博之. “日本語意味解析システム SAGE の高速化・高精度化とコーパスによる精度評価,” 情報処理学会論文誌, Vol 43, No.9, pp.2894-2902.

[15] 井村 裕, 沓掛 俊樹, 佐藤 直美, 原田 実. “意味解析システム SAGE の Web 化と連体・使役・受身における解析精度向上,” 人工知能学会第 14 回全国大会論文集, pp.149-152 (2003).

[16] Kadri Hacioglu, Wayne Ward, “Target Word Detection and Semantic Role Chunking using Support Vector Machines,” Proceedings of the Human Language Technology

Conference (2003).

[17] Tom O'Hara, Janyce Wiebe, "Preposition Semantic Classification via TREEBANK and FRAMENET," submitted to 41st Annual Meeting of the Association for Commputational Linguistics (2003).

[18] Sameer Pradhan, Kadri Hacioglu, Wayne Ward, James H. Martin, Daniel Jurafsky, "Semantic Role Parsing: Adding Semantic Structure to Unstructured Text," Proceedings of the International Conference on Data Mining (ICDM-2003), Melbourne, FL, Nov. 19-22 (2003).

[19] Daniel Gildea, Daniel Jurafsky, "Automatic Labeling of Semantic Roles," Computational Linguistics 28(3), pp245-288 (2002).

付録

付録１　EDR コーパス: 概念関係子

関係子	意味
agent	有意志動作を引き起こす主体
object	動作・変化の影響を受ける対象
a-object	属性を持つ対象
implement	有意志動作における道具・手段
material	材料または構成要素
source	事象の主体または対象の最初の位置
goal	事象の主体または対象の最後の位置
place	事象の成立する場所
scene	事象の成立する場面
basis	比較の基準
manner	動作・変化のやり方
time	事象の起こる時間
time-from	事象の始まる時間
time-to	事象の終わる時間
quantity	物・動作・変化の量
modifier	修飾関係
number	数
and	概念間の連結関係
or	概念間の選択関係
condition	事象・事実の条件関係
purpose	目的
cooccurrence	事象・事実の同時関係
sequence	事象・事実の時間的前後関係
possessor	所有関係
beneficiary	利益・不利益の移動先
unit	単位
from-to	範囲

付録２　EDR コーパス: 事象・事実表現のための概念属性子

属性子	意味
not	否定
generic	総称
all	全て
some	任意
each	各々
this	近指示（その）
that	遠指示（あの）
specific	特定のインスタンス

付録３　EDR コーパス: 話者の視点がある時点を表す概念属性子

属性子	意味
past	視点が過去
present	視点が現在
future	視点が未来

付録４　EDR コーパス: 相情報を表すための概念属性子

属性子	意味
bigen	動作や現象が開始するということを表す
end	動作や現象が終了したということを表す
progress	動作や現象が開始してからまだ終わっていないということを表す
continue	反復動作が継続中であることを表す
state	動作が終了し、達成された状態や結果が残っているということを表す
yet	まだ開始していない状態、または終了していないを示す
already	すでに開始した、または終了した状態を示す
soon	間もなく開始する、または終了する状態を示す
just	ほんの少し前に開始した、または終了した状態を示す
complete	目的とした動作の全てを完了するということを表す
come	話者の思っている基準点に近づく
go	話者の思っている基準点から離れる

付録５　EDR コーパス: 文要素に関する話者の意図・判断を表す概念属性子

属性子	意味
imperative	命令（～せよ）
grant	許可（～してもよい、～で構わない、～ても構わない）
consent	同意（～してもよい、～で構わない、～ても構わない）
grant-not	禁止（～してはならない）
advise	話者が相手に忠告・勧告をする表現（～することだ）
recommend	推薦（～した方がよい、～する事がよい）
invite	勧誘（～しよう）
require-agreement	同意・確認を求める行為（～ですね）
polite	丁寧（どうぞ～下さい）
respect	尊敬（～れる、～られる、敬語、尊敬語）
should	義務（～すべきだ）
sufficiency	十分（～ればよい、～だけでよい）
duty	義務（～しなくてはならない、～しなければならない）
interrogation	疑問
conclude	断定（～したのだ、～である）
sure	状況からの推量・確信（～するに違いない、～のはずだ）
maybe	可能性があると思っている水量（～するかもしれない）
seem	推察・推測（～するだろう、～そうだ、～らしい）
rumor	伝聞（～だそうだ、～するらしい、～ようだ）
appearance	様子・比況（～にみえる、～みたい）
be-sorry	後悔（～したかったのに）
natural-result	当然の帰結（～するわけだ）
natural-thing	理想の姿・当為（～ものだ）
if	不確実なことの家庭
thought	事実に反対のこと（もし～だったら）
reality	事実として
exclamation	感嘆
pity	哀れみ
blame	非難
unexpected	予想外・意外
underestimate	過小評価（～に過ぎない、～しただけである）

付録 6　YamCha コマンドのオプション

オプション省略形	オプション	説明
-m	**--model=*FILE***	**モデルファイルを指定する**
-F	**--feature=*PAT***	**特徴としてパターン *PAT を使用する***
-e	**--eos-string=*STR***	**文の区切り文字として *STR を使う***
-V	**--verbose**	**verbose モードにする**
-C	**--candidate**	**部分 Chunking を行う**
-B	**--backward**	**文の後ろから判定を行う**
-o	**--output=*FILE***	**出力ファイルを指定する**
-v	**--version**	**バージョンを出力する**
-h	**--help**	**ヘルプを出力する**

付録 7　文中における 1 組の深層格組の共起領域