自然言語処理 140－2
（2000．11．21）

Support Vector Machine を用いた Chunk 同定

工藤 拓　　松本 裕治

奈良先端科学技術大学院大学 情報科学研究科

〒 630-0101 奈良県生駒市高山町 8916-5

{taku-ku,matsu}@is.aist-nara.ac.jp

本稿では, Support Vector Machine (SVM) に基づく一般的な chunk 同定手法を提案し, その評価を行なう. SVM は従来からある学習モデルと比較して, 入力次元数に依存しない極めて高い汎化能力を持ち, Kernel 関数を導入することで効率良く素性の組み合わせを考慮しながら分類問題を学習することが可能である. SVM を英語の単名詞句とその他の句の同定問題に適用し, 実際のタグ付けデータを用いて解析を行なったところ, 従来手法に比べて非常に高い精度を示した. さらに, chunk の表現手法が異なる複数のモデルの重み付き多数決を行なうことでさらなる精度向上を示すことができた.

キーワード: Chunking, 文節まとめ上げ, 機械学習, Support Vector Machine, 重み付き多数決

Chunking with Support Vector Machines

Taku Kudoh　　Yuji Matsumoto

Graduate School of Infomation Science, Nara Institute Science and Technology

8916-5 Takayama, Ikoma Nara 630-0101 Japan

{taku-ku,matsu}@is.aist-nara.ac.jp

In this paper, we apply Support Vector Machines (SVMs) to identify English base phrases (chunks). It is well-known that SVMs achieve high generalization performance even with input data of very high dimensional feature space. Furthermore, by introducing the Kernel principle, SVMs can carry out the training in a high-dimensional space with smaller computational cost independent of their dimensionality. In order to achieve higher accuracy, we also apply majority voting of 8 SVM-based systems which are trained using distinct chunk representations. Experimental results show that our approach achieves better accuracy than other conventional frameworks.

Keywords : Chunking, Machine Learning, Support Vector Machines, Majority Voting

1 はじめに

自然言語処理において chunk 同定問題 (chunking) とは, 任意の token をある視点からまとめ上げていき, まとめ上げた固まり (chunk) をそれらが果たす機能ごとに分類する一連の手続きのことを示す. この問題の範疇にある処理として, 英語の単名詞句同定 (base NP chunking), 任意の句の同定 (chunking), 日本語の文節まとめ上げ, 固有名詞/専門用語抽出などがある. また, 各文字を token としてとらえるならば, 英語の tokenization, 日本語のわかち書き, 品詞タグ付けなども chunk 同定問題の一部としてとらえることができる.

一般に, chunk 同定問題は, 各文脈を素性としてとらえ, それらの情報から精度良く chunk を同定するルールを導出する手続きとみなす事ができるため, 各種の統計的機械学習アルゴリズムを適用することが考えられる. 実際に機械学習を用いた多くの chunk 同定手法が提案されている [8, 9, 12, 10, 20, 14, 19, 17].

しかしながら, 従来の統計的手法は, 潜在的に多くの問題をかかえている. 例えば, 隠れマルコフモデルや Maximum Entropy (ME) モデルは素性どうしの組み合わせを効率良く学習できず, 有効な組み合わせの多くは人間の発見的な手続きで決定されている. また多く機械学習アルゴリズムは高い精度を得るために慎重な素性選択を要求し, これらの素性選択も人間の発見的な手続きにたよっている場合が多い.

一方, 統計的機械学習の分野では, Support Vector Machine (SVM) [3, 18] 等の学習サンプルと分類境界の間隔 (マージン) を最大化にするような戦略に基づく手法が提案されている. 特に SVM は, 学習データの次元数 (素性集合) に依存しない極めて高い汎化能力を持ち合わせている事が実験的にも理論的にも明らかになっている. さらに, Kernel 関数を導入するとこで, 非線形のモデル空間を仮定したり, 複数の素性の組み合せを考慮した学習が可能である.

このような優位性から, SVM は多くのパターン認識の分野に応用されている. 自然言語処理の分野においても, 文書分類や係り受け解析に応用されており, 従来の手法に比べて高い性能を示している [6, 4, 21].

本稿では chunk 同定問題として, 英語の単名詞句のまとめ上げ (base NP chunking) および英語の任意の句の同定 (chunking) を例にとりながら学習手法として SVM を用いた手法を述べる. さらに, chunk の表現方法が異なる個々の学習データから独立に学習し, それらの重み付け多数決を行なう事でさらなる精度向上を試みる. その際, 本稿では, 各モデルの重みとして SVM に固有の新たな 2 種類の重み付けの手法を提案する.

本稿の構成は以下の通りである. 2 章で SVM の概要を説明し, 3 章で一般的な chunk 同定モデルおよび SVM の具体的な適用方法, 重み付け多数決の方法について述べる. さらに 4 章で実際のタグ付きコーパスを用いた評価実験を提示し, 最後に 5 章で本稿をまとめる.

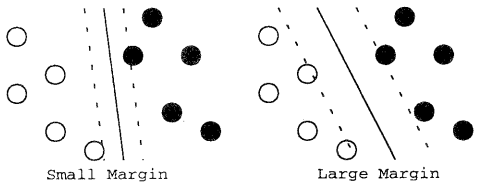

図 1: マージン最大化

2 Support Vector Machine

2.1 Optimal Hyperplane

正例, 負例 の 2 つのクラスに属す学習データのベクトル集合を,

$$(\mathbf{x}_i, y_i), \ldots, (\mathbf{x}_l, y_l) \qquad \mathbf{x}_i \in \mathbf{R}^n,\ y_i \in \{+1, -1\}$$

とする. ここで $\mathbf{x}_i$ はデータ i の特徴ベクトルで, 一般的に n 次元の素性ベクトル ($\mathbf{x}_i = (f_1, f_2, \ldots, f_n) \in \mathbf{R}^n$) で表現される. y_i はデータ i が, 正例 (1), 負例 (-1) かを表わすスカラーである. パターン認識とは, この学習データ $\mathbf{x}_i \in \mathbf{R}^n$ から, クラスラベル出力 $y \in \{\pm 1\}$ への識別関数 $f : \mathbf{R}^n \to \{\pm 1\}$ を導出することにある.

SVM では, 以下のような n 次元 Euclid 空間上の平面で正例, 負例を分離することを考える.

$$\mathbf{w} \cdot \mathbf{x} + b = 0 \qquad \mathbf{w} \in \mathbf{R}^n, b \in \mathbf{R} \tag{1}$$

この時, 近接する正例と負例の間の間隔 (マージン) ができるだけ大きいほうが, 汎化能力が高く, 精度よく評価データを分類できる. 図 1 に, 2 次元空間上の正例 (白丸), 負例 (黒丸) を分離する問題を例にこのマージン最大化の概略を表す. 図 1 中の実線は式 (1) の分離平面を示す. 一般にこのような分離平面は無数に存在し, 図 1 に示す 2 つの分離平面はどちらも学習データを誤りなく分離している. 分離平面に平行する 2 つの破線は分離平面が傾き $\mathbf{w}$ を変化させないまま平行移動したときに, 分類誤りなく移動できる境界を示す. この 2 つの破線間の距離をマージンと呼び, SVM はマージンが最大となる分離平面を求める戦略を採用している. 図 1 の例では, 右の分離平面が左の分離平面にくらべて大きなマージンを持っており, 精度よくテスト事例を分離できることを意味している.

実際に 2 つの破線を求めてみる. 破線は, 正例 (+1) もしくは負例 (-1) のラベルを出力する境界面である

と考えれば,

$$\mathbf{w}\cdot\mathbf{x}+b=\pm 1 \qquad \mathbf{w}\in\mathbf{R}^n, b\in\mathbf{R}$$

で与えられる. さらにマージン d は, 分離平面上の任意の点 $\mathbf{x}_i$ から二つの破線までの距離を考慮すれば,

$$d=\frac{|\mathbf{w}\cdot\mathbf{x}_i+b-1|}{\|\mathbf{w}\|}+\frac{|\mathbf{w}\cdot\mathbf{x}_i+b+1|}{\|\mathbf{w}\|}=\frac{2}{\|\mathbf{w}\|}$$

となる. このマージンを最大化するためには, $\|\mathbf{w}\|$ を最小化すればよい. つまり, この問題は以下の制約付き最適化問題を解くことと等価となる[1].

$$\text{目的関数：}\quad L(\mathbf{w})=\frac{1}{2}\|\mathbf{w}\|^2 \rightarrow \text{最小化}$$

$$\text{制約条件：}\quad y_i[(\mathbf{w}\cdot\mathbf{x}_i)+b]\geq 1\ (i=1\ldots l)$$

ここで, 2 つの破線上の分類を決定づける事例を support vector と呼び, support vector 以外の事例は実際の学習結果に影響を及ぼさない.

さらに, 一般的な分類問題においては, 学習データを線形分離することが困難な場合ある. このような場合, 各素性の組み合わせを考慮し, より高次元な空間に学習データを写像すれば線形分離が容易になる. 実際の証明は省略するが SVM の学習, 分類アルゴリズムは各事例間の内積のみしか使用しない. この点を生かし, 各事例間の内積を任意の Kernel 関数におきかえることで, SVM は低次元中の非線形分類問題を高次元中の線形分離問題としてみなし分類を行なう事が可能となっている. 多くの Kernel 関数が提案されているが, 我々は以下の式で与えられる d 次の Polynomial Kernel 関数を用いた.

$$K(\mathbf{x},\mathbf{y})=(\mathbf{x}\cdot\mathbf{y}+1)^d$$

d 次の Polynomial 関数は d 個までの素性の組み合わせ (共起) を考慮した学習モデルに帰着できる.

2.2 SVM の汎化能力

ここで, 汎化能力に関する一般的な理論について考察してみる. 学習データおよびテストデータがすべて独立かつ同じ分布 $P(\mathbf{x},y)$ から生成されたと仮定すると, 識別関数 f のテストデータに対する汎化誤差 $E_g[f]$, 学習データに対する誤差 $E_t[f]$ は以下のように与えられる.

$$E_g[f] = \int \frac{1}{2}|f(\mathbf{x})-y|dP(\mathbf{x},y)$$

$$E_t[f] = \frac{1}{l}\sum_{i=1}^{l}\frac{1}{2}|f(\mathbf{x}_i)-y_i)|$$

さらに, $E_g[f], E_t[f]$ には以下のような関係が成立することが知られている [18].

[1]実際の我々の実験では多少の解析誤りを認める Soft Margin の項を追加した最適化問題を解いている

定理 1 (Vapnik) 学習データの事例数を l, モデルの VC次元を h とする時, 汎化誤差 $E_g[f]$ は, $1-\eta$ の確率で以下の上限値を持つ.

$$E_g[f]\leq E_t[f]+\sqrt{\frac{h(\ln\frac{2l}{h}+1)-\ln\frac{\eta}{4}}{l}} \qquad (2)$$

ここで VC 次元 h とは, モデルの記述能力, 複雑さを表すパラメータである. 式 (2) の右辺を VC bound と呼び, 汎化誤差を小さくするには, VC bound をできるだけ小さくすればよい.

従来からる多くの学習アルゴリズムは, モデルの複雑さである VC 次元 h を固定し, 学習データに対するエラー率を最小にするような戦略をとる. そのため, 適切に h を選ばないとテストデータを精度良く分類できない. また適切な h の選択は一般的に困難である.

一方 SVM は, 学習データに対するエラー率を Soft Margin や Kernel 関数を使って固定し, そのうえで右辺の第二項を最小化する戦略をとる. 実際に式 (2) の右辺第二項に注目すると, h に対して増加関数となっている. つまり, 汎化誤差 $E_g(h)$ を小さくするには, h をできるだけ小さくすればよい. SVM では VC 次元 h とマージン M には以下の関係が成立することが知られている [18].

定理 2 (Vapnik) 事例の次元数を n, マージンを M, 全事例を囲む球面の最小直径を D とすると, SVM の VC次元 h は, 以下の上限値を持つ

$$h\leq \min(D^2/M^2, n)+1 \qquad (3)$$

式 (3) から, h を最小にするためには, マージンを最大にすればよく, これは SVM がとる戦略そのものである事が分かる. また, 学習データの次元数が十分大きければ, VC 次元 h は, 学習データの次元数に依存しない. さらに, D は, 使用する Kernel 関数によって決まるため, 式 (3) は Kernel 関数の選択の指針を与える能力も持ちあわせていることが知られている [18]. また, Vapnik は式 (2) とは別に, SVM に固有のエラー率の上限を与えている.

定理 3 (Vapnik) $E_l[f]$ を *Leave-One-Out* によって評価されるエラー率とする場合

$$E_l[f]\leq \frac{Number\ of\ Support\ Vectors}{Number\ of\ training\ samples} \qquad (4)$$

となる.

Leave-One-Out とは, l 個の学習データのうち 1 個をとりのぞいてテストデータとし, 残り $l-1$ を使って学習することをすべてのデータについて l 回くりかえすことを指す. 式 (4) は容易に証明可能である. つまり, SVM の特徴として support vector 以外の事例

は最終の識別関数には一切影響を及ぼさない．そのため個々の support vector すべてが誤ったときが最悪のケースとなり，式 (4) が導かれる．この bound は，単純明解で汎化誤差のおおまかな値を予測することを可能にする．しかし，support vector の数が増えても汎化能力が向上する事例もあり，式 (4) の汎化誤差の予測能力は式 (2) にはおとる事が知られている．

3 SVM に基づく Chunk 同定

3.1 Chunk の表現方法

chunk 同定の際，各 chunk の状態をどう表現するかが問題となる．一つの手法として，各 chunk 同定を分割問題とみなし，各単語の間 (ギャップ) にタグを付与する手法が考えられる．しかし，この手法は単語とは別の位置にタグを付与する必要があり，従来からある形態素解析などのタグ付けタスクとは異なる枠組が必要となる．

その一方で，各単語に chunk の状態を示すタグを付与する手法がある．この手法は従来からあるタグ付け問題と同じ枠組でモデル化ができる利点がある．

後者の単語にタグを付与する表現法として，以下 2 種類の手法が提案されている．

1. Inside/Outside
   この手法は英語の base NP 同定でよく用いられる手法の一つである [8]．この手法では，chunk の状態として以下の 3 種類を設定する．

   I　現在位置の単語は chunk の一部である．
   O　現在位置の単語は chunk に含まれない．
   B　現在位置の単語はある chunk の直後に位置する chunk の先頭である．

   さらに Tjong Kim Sang らは，上記のモデルを IOB1 と呼び，このモデルを基に IOB2/IOE1/IOE2 の 3 種類の表現方法を提案している [13]．

   IOB2　IOB1 と基本的に同じだが，B タグの意味づけがことなる．IOB2 の場合，B タグはすべての chunk の先頭に付与される．
   IOE1　IOB1 と基本的に同じだが，B タグの代わりに E タグを導入する．E タグは，ある chunk の直前に位置する chunk の末尾の単語に付与される．
   IOE2　IOE1 と基本的に同じだが，E タグはすべての chunk の末尾の単語に付与される．

2. Start/End
   この手法は日本語固有名詞抽出において用いられた手法 [20] で，各単語に付与するタグとして以下の 5 種類を設定する[2]．

   B　現在位置の単語は，一つ以上の単語から構成される chunk の先頭の単語である．
   E　現在位置の単語は，一つ以上の単語から構成される chunk の末尾の単語である．
   I　現在位置の単語は，一つ以上の単語から構成される chunk の先頭，末尾以外の中間の単語である．
   S　現在位置の単語は 単独で一つの chunk を構成する．
   O　現在位置の単語は chunk に含まれない

これら 5 種類のタグ付け手法を base NP chunking を例に以下に示す．

	IOB1	IOB2	IOE1	IOE2	IOBES
In	O	O	O	O	O
early	I	B	I	I	B
trading	I	I	I	E	E
in	O	O	O	O	O
busy	I	B	I	I	B
Hong	I	I	I	I	I
Kong	I	I	E	E	E
Monday	B	B	I	E	S
,	O	O	O	O	O
gold	I	B	I	E	S
was	O	O	O	O	O

各 chunk に対し，その chunk の役割を示すタグを付与する場合は，B/E/I/O/S といった chunk の状態を示すタグと，役割を示すタグを '-' で連結し新たなタグを導入することによって表現する．例えば，IOB2 モデルにおいて，動詞句 (VP) の先頭の単語は B-VP というタグが付与される．

3.2 SVM による Chunk 同定

現在の位置の単語に対して chunk タグを付与する際に，周囲の単語や品詞等の文脈素性をどう選択するかが問題となる．

本稿では，一般的に用いられる固定長モデルを採用した．具体的には，位置 i の chunk タグ c_i の推定を行なう素性として c_i 自身の単語と品詞，および右 2 つ，左 2 つの単語と品詞を用いた．さらに左 2 つの chunk タグも素性として使用した．

[2]内元らは,C/E/U/O/S の 5 種類のタグを用いているが，IOB1/IOB2/IOE1/IOE2 モデルとの整合性から，便宜的に B/E/I/O/S タグを用いる．タグの名称の変更のみで本質的なタグの意味づけに変更はない．

		→	解析方向	→	
単語:	w_{i-2}	w_{i-1}	w_i	w_{i+1}	w_{i+1}
品詞:	t_{i-2}	t_{i-1}	t_i	t_{i+1}	t_{i+1}
chunk:	c_{i-2}	c_{i-1}			

一般に, 左 2 つの chunk タグは学習データに対しては付与されているが, テストデータに対しては付与されていない. そこで実際の解析時には, これらは左から右向きに解析しながら動的に追加していくこととした.

さらに, 解析方向を逆 (右向きから左向き) にし, 右 2 つの chunk を素性として使用することも考えられる. 本稿では, これら 2 つの解析手法を前方向解析/後ろ方向解析と呼び区別する.

基本的に SVM は 2 値分類器である. そのため, chunk のタグ表現のように多値の分類問題を扱うためには SVM に対し何らかの拡張を行なう必要がある. 一般に, 2 値分類器を多値分類器に拡張する手法として, 以下に述べる 2 種類の手法がある. 一つは, *"one class vs. all others"* と呼ばれる手法で, K クラスの分類問題に対し, あるクラスかそれ以外かを分類する計 K 種類の分類器を作成する手法である. もう一つは, *"pairwise classification"* であり, 各クラス 2 つの組み合わせを分類する $K \times (K-2)/2$ 種類の分類器を作成し, 最終的にそれらの多数決でクラスを決定する手法である.

本稿では, 後者の *"pairwise classification"* を採用した. その理由として, (1) 各分類器の学習に用いられる学習データが少量であり, 学習のコストが小さいこと, (2) *"one class vs. all others"* よりも *"pairwise classification"* が実験的に良い結果が得られたという報告 [5] があること, の 2 点が挙げられる.

3.3 重み付き多数決

Tjong Kim Sang らは, base NP 同定の問題に対し, 弱学習アルゴリズムに MBL, ME, IGTree 等の 7 種類のアルゴリズム, さらに IOB1/IOB2/IOE1/IOE2 の 4 種類の表現を用いて独立に学習した複数のモデルの重み付き多数決を行うことで, 個々のモデルのどれよりも高精度の結果が得られたと報告している [9, 12].

このような重み付き多数決の手法は, 潜在的にマージン最大化の効果があり, 汎化能力の高い強学習アルゴリズムが作成できる事が理論的にも実験的にも明らかになっている [15]. この重み付き多数決の概念の一つとして Boosting があり, 文書分類や日本語の係り受け解析に応用され非常に高い精度を示している.

本稿では, 弱学習アルゴリズムに SVM を用い, IOB1/IOB2/IOE1/IOE2 の 4 種類の表現, さらに解析方向 (前方向/後ろ方向) の合計 $4 \times 2 = 8$ 種類の重み付け多数決を行なう事でさらなる精度向上を試みる.

重み付き多数決を行なう場合, 各モデルの重みをどう決定するかが問題となる, 真のテストデータに対する精度を用いることで最良の結果を得ることが出来るが, 一般に真のテストデータを評価することは不可能である. Boosting では学習データの頻度分布を変更しながら, 各ラウンドにおける学習データに対する精度を重みとしている. しかしながら, SVM は, Soft Margin パラメータ, Kernel 関数の選択次第で, 学習データは完全に分離する事ができ, 単純に学習データに対する精度を重みにするすことは困難である. また, Boosting のように学習データの頻度分布を変化させても, SVM は学習データの頻度には一切依存しない学習アルゴリズムであるために精度は変化しない.

本稿では, 重み付き多数決の重みとして以下の 4 種類の手法を提案し, それぞれの手法の精度や計算量などを考察する.

1. **均一重み**
   これは, すべてのモデルに対し均一の重みを付与する手法である. 最も単純な手法であり, 他の手法に対するベースラインとなる.
2. **交差検定**
   学習データを N 等分し, $N-1$ を学習データ, 残りの 1 をテストとして評価する. この処理を N 回行ない, それぞれの精度の平均を各モデルの重みとして利用する.
3. **VC bound**
   式 (3), 式 (2) を用いて VC bound を計算し, その値から正解率の下限を推定し[3], 重みとする手法である. ただし, 式 (3) における全事例を囲む最小直径 D は以下のように各学習データの最大ノルムで近似を行なう.

   $$D^2 \sim \max_i(\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_i)) = \max_i K(\mathbf{x}_i, \mathbf{x}_i)$$

4. **Leave-One-Out bound**
   式 (4) の Leave-One-Out bound を求め, 正解率の下限を推定し, 重みとする手法である.

実際の解析は以下のように行なわれる.

1. 学習データを IOB1/IOB2/IOE1/IOE2 の各表現に変換する.
2. 4 つの表現に対し, 前方向解析, 後ろ方向解析の計 $4 \times 2 = 8$ 種類のモデルを作成し, SVM で独立に学習する.
3. 8 種類のモデルに対し, VC bound, Leave-One-Out bound を計算し重みを求める. 交差検定に関しては, 1,2 の処理を各分割したデータに対して行ない, 各ラウンドのタグ付け精度の平均を重みとする.

[3] エラー率の上限であるため, 1 からこの値を引き, 正解率の下限とみなす.

4. 合計 8 種類のモデルを用いてテストデータを解析する. 解析後のデータをらにIOB1/IOB2/IOE1/IOE2 の各表現に変換する. つまり最終的に計 8(各モデル)×4(変換先) = 32 種類の結果を得ることとなる.

5. IOB1/IOB2/IOE1/IOE2 の個々対し, タグレベルで合計 8 種類の重みつき多数決を行う[4]. つまり各重み付けの手法に対し, IOB1/IOB2/IOE1/IOE2 の 4 種類の表現方法で評価した結果を得ることとなる.

4 実験と考察

4.1 実験環境, 設定

実験には以下の 3 種類のタグ付きデータを用いた.

- base NP 標準データセット (**baseNP-S**)
  Penn Tree-bank/WSJ の 15-18 を学習データ, 21 をテストデータとし, Brill Tagger[1] を用いて part-of-speech (POS) を付与したデータである これは base NP 同定の標準データセットとして認識されている[5].
- base NP Large データセット (**baseNP-L**)
  Penn Tree-bank/WSJ の 02-21 を学習データ, 00 をテストデータとし, Brill Tagger を用いて POS を付与したデータである. ただし, IOB1/IOE1 の表現でタグ付けを行なう場合, I/O のタグ数が多くなり, プログラムの制約上から実験を行なうことができなかった. そのため このデータセットに限り IOB2/IOE2 のみの評価を行なった. また, 計算時間上の問題から, 交差検定による重みの推定は行なっていない.
- Chunking データセット (**chunking**)
  CoNLL-2000 の Shared Task[11] に用いられたデータである. base NP 標準データセットと基本的に同じだが, base NP 以外に VP, PP, ADJP, ADVP, CONJP, INITJ, LST, PRT, SBAR の合計 10 種類の英語の句を表現するタグが付与されている[6].

実験には自作の SVM 学習ツール TinySVM を用いた[7]. このツールは, 本実験のようなバイナリの素性表現に特化して高速化を試みたツールであり, 同時に VC bound を自動的に推定する機能を持っている. さらに, すべての実験において, Kernel 関数は 2 次の Polynomial 関数, Soft Margin は 1 に固定した.

[4]実際には chunk レベルで行なわないと chunk の整合性が取れなくなる可能性があるが, 本稿では問題を簡単にするため タグレベルで多数決を取ることとした

[5]ftp://ftp.cis.upenn.edu/pub/chunker/から入手可能

[6]http://lcg-www.uia.ac.be/conll2000/chunking/ から入手可能

[7]2000 年 11 月現在, 松本研究室内で利用可能, 一般公開の予定あり

学習条件		精度	推定重み		
学習データ	変換先	$F_{\beta=1}$	B	C	D
baseNP-S	IOB1-前	93.76	.9394	.4310	.9193
	IOB1-後	93.93	.9422	.4351	.9184
	IOB2-前	93.84	.9410	.4415	.9172
	IOB2-後	93.70	.9407	.4300	.9166
	IOE1-前	93.73	.9386	.4274	.9183
	IOE1-後	93.98	.9425	.4400	.9217
	IOE2-前	93.98	.9409	.4350	.9180
	IOE2-後	**94.11**	.9426	.4510	.9193
baseNP-L	IOB2-前	**95.34**	-	.4500	.9497
	IOB2-後	95.28	-	.4362	.9487
	IOE2-前	95.32	-	.4467	.9496
	IOE2-後	95.29	-	.4556	.9503
chunking	IOB1-前	93.48	.9342	.6585	.9605
	IOB1-後	93.74	.9346	.6614	.9596
	IOB2-前	93.46	.9341	.6809	.9586
	IOB2-後	93.47	.9355	.6722	.9594
	IOE1-前	93.45	.9335	.6533	.9589
	IOE1-後	93.72	.9358	.6669	.9611
	IOE2-前	93.45	.9341	.6740	.9606
	IOE2-後	**93.85**	.9361	.6913	.9597
baseNP-S	IOBES-前	93.96			
	IOBES-後	93.58			
chunking	IOBES-前	93.31			
	IOBES-後	93.41			

B:交差検定 C:VC bound D:Leave-One-Out bound

表 1: 個々のモデルの精度比較

評価方法としては, 適合率と再現率の調和平均で与えられる F 値 ($\beta = 1$) を用いた. これは chunk 同定において一般的に用いられる評価方法である. 以後特にことわりが無い限り F 値のことを精度と呼ぶ.

4.2 実験結果

表 1 に, 各 chunk の表現方法, および解析方向が異なる計 8 種のモデルで独立に学習した実験結果 (テストデータに対する精度, 推定された重み) をまとめた. また, 比較対象として, Start/End 法を用いた学習結果についても掲載している.

さらに, 表 2 に, これらを A:均一重み, B:交差検定 ($N = 5$), C:VC vound, D:Leave on Out bound の 4 種類の重み付けで多数決を行なった際の結果をまとめた. 表 3 には, 各の重み付け手法の中の最良の結果について, その適合率と正解率を示した.

4.3 Chunk の表現方法と解析精度

表 1 から, baseNP-L データセットを除き, IOE2+後ろ向き解析が比較的良い精度を示している. また, Inside/Outside 法 (IOB1/IOB2/IOE1/IOE2 の各手法) と Start/End 法の精度を比較してみても, 顕著な

学習条件		各重み付けに対する精度 $F_{\beta=1}$			
学習データ	評価手法	A	B	C	D
baseNP-S	IOB1	94.14	94.20	94.20	94.16
	IOB2	94.16	**94.22**	**94.22**	94.18
	IOE1	94.14	94.19	94.19	94.16
	IOE2	94.16	94.20	94.21	94.17
baseNP-L	IOB2	**95.77**	-	95.66	95.66
	IOE2	**95.77**	-	95.66	95.66
chunking	IOB1	93.77	93.87	93.89	93.87
	IOB2	93.72	93.87	93.90	93.88
	IOE1	93.76	93.86	93.88	93.86
	IOE2	93.77	93.89	**93.91**	93.85

A:均一 B:交差検定 C:VC bound D:Leave-One-Out bound

表 2: 重み付き多数決の結果

データセット	適合率	再現率	$F_{\beta=1}$
baseNP-S	94.15%	94.29%	94.22
baseNP-L	95.62%	95.93%	95.77
chunking	93.89%	93.92%	93.91

表 3: 各データセットに対する最良結果

精度差はみられなかった. Sassano らは, 各学習アルゴリズムの特徴を考察しながら, 決定リストは細かい組み合せを考慮する Start/End 法が, ME はより粗い情報を考慮する Inside/Outside 法が精度が良いと報告している [19]. SVM においてこれらの二つの手法に顕著な差が見られないのは, 細かい情報から粗い情報を極めて柔軟に選択し, 最良の分離平面を構築しているからではないかと考える.

4.4 多数決の効果

表 2 から多数決を行なうことでその重みの付与方法によらず, 個々のどのモデルよりも精度が向上することが確認できる. また, baseNP-L データセット以外は, 交差検定, VC bound, Leave one Out bound ともにベースラインである均一の重み付けより精度が向上している.

VC bound を用いた場合は, 交差検定とほぼ同等の結果を得ることができた. これは VC bound が真のテストデータに対する精度を極めて精度よく推定する能力を持ち合わせていることを示唆している. 実際に表 1 において, テストデータに対する精度と推定された各重みの関係を見ると, VC bound は予測値と精度との差は大きいが, テストデータに対する精度を精度よく予測している事が分かる. 逆に, Leave-One-Out bound は, それ自身の予測能力は VC bound におとる事が分かる.

交差検定はこのような複数モデルの重み付けを行なう際によく用いられる手法であるが, 分割数が多くなると精度を推定するのに莫大な計算量を必要とする. その一方で, VC bound は一回の学習で非常に精度の良い重みを推定する事ができるため交差検定に比べ効率的であると考える.

4.5 関連研究との比較

Tjong Kim Sang らは, 弱学習アルゴリズムに MBL, ME, IGTree 等の 7 種類のアルゴリズム, さらに IOB1/IOB2/IOE1/IOE2 の 4 種類の表現を用いて独立に学習した複数のモデルの重み付き多数決を行うことで, baseNP-S データセットに対し 93.86, baseNP-L に対し 94.90 の精度が得られたと報告している [9, 12].

我々は単独の表現を用いた場合でも 93.76 - 94.11 (baseNP-S), 95.29 - 95.34 (baseNP-L), さらに各表現の重み付き多数決を行なう事で 94.22 (baseNP-S), 95.77 (baseNP-L) の精度を得ている. 単純な精度比較においては, SVM は MBL, ME, IGTree といった従来のアルゴリズムのどれよりも高性能であると我々は考える.

CoNLL-2000 Shared Task において我々は SVM と単独のタグ表現を用いて 93.48 の精度を報告した [7]. 本稿で提案する重み付き多数決を行なうことで 93.91 の精度を示すことができ, 単独のタグ表現を用いる手法を上回る結果となった. また CoNLL-2000 で報告された重み付き多数決に基づく他の手法 [17, 10] よりも高い精度を示すことができた.

内元らは, ME に基づくモデルを用い, 日本語の固有名詞抽出を行っている [20]. しかし, ME はすべての素性を独立として扱うため, それぞれの素性の組合わせを考慮する場合は, 組合わせを新しい素性として追加する必要がある. 内元らは, 重要だと思われる素性の組み合わせを人手により発見的に選択しているが, 必ずしも重要な素性の組み合わせを網羅しているとは限らない. SVM は, Kernel 関数の変更という操作のみで, 計算量をほとんど変えることなく組み合わせを含めた学習が行なえるため, 網羅性, 一貫性という意味で優位であると考える.

4.6 今後の課題

- 他の分野への応用
  我々の提案する手法は, 日本語の文節まとめ上げや固有名詞, 専門用語抽出と一般的な chunk 同定問題に応用可能である. 我々の提案する手法がこれらの他の分野でも有効であるか実際に検証を行なう予定である.
- 可変長モデル
  本稿では, 左右 2 つの文脈のみを考慮する単純な固定長モデルを採用した. しかし実際には, 個々の chunk を同定に必要な文脈長は可変であり,

個々の chunk に対し最適な文脈長を選択することでさらなる精度向上が期待できる. Sassano らは日本語の固有名詞抽出において可変長モデルを提案し単純な固定長のモデルより高い精度が得られたと報告している [14, 19]. 我々は, Virtual SVM[16] のアイデアを取りいれ, 抽出される support ector に対し変形規則を適用し, 可変長の文脈を間接的に考慮するモデルを構築したいと考えている.

- より予測能力の高い bound の採用
  本稿では, 重み付き多数決の重みとして, SVM に固有の概念 — VC bound, Leave-One-Out bound を提案した. その一方で Chapelle らは, これらより予測能力の高い bound を提案し,Kernel 関数の選択や Soft Margin パラメータの選択に極めて有効であるとこを示している [2]. これらの予測能力の高い bound を重みとして採用することでさらなる精度向上が期待できるのではないだろうか.

5 まとめ

本稿では, Support Vector Machine (SVM) に基づく一般的な chunk 同定問題の解析手法を提案し, 実際のタグ付きコーパスを使用して実験を行なった. 英語 chunking における実験では, 過去の MBL や ME に基づくモデルよりも高い精度を示し, SVM の持つ高い汎化能力を裏づける結果となった. さらに, chunk の表現方法が異なるデータを使い独立に学習し, それらの重み付き多数決を行なう事で, 個々のどのモデルよりも高い精度を示すことができた.

参考文献

[1] Eric Brill. Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part-of-Speech Tagging. *Computational Linguistics*, Vol. 21, No. 4, 1995.

[2] Oliver Chapelle and Vladimir Vapnik. Model selection for support vector machines. In *Advances in Neural Information Processing Systems 12*. Cambridge, Mass: MIT Press, 2000.

[3] C. Cortes and Vladimir N. Vapnik. Support Vector Networks. *Machine Learning*, Vol. 20, pp. 273–297, 1995.

[4] Thorsten Joachims. Transductive Inference for Text Classification using Support Vector Machines. In *International Conference on Machine Learning (ICML)*, 1999.

[5] Ulrich H.-G Kreßel. Pairwise Classification and Support Vector Machines. In *Advances in Kernel Mathods*. MIT Press, 1999.

[6] Taku Kudoh and Yuji Matsumoto. Japanese Dependency Structure Analysis Based on Support Vector Machines. In *Empirical Methods in Natural Language Processing and Very Large Corpora*, pp. 18–25, 2000.

[7] Taku Kudoh and Yuji Matsumoto. Use of Support Vector Learning for Chunk Identification. In *Proceedings of the 4th Conference on CoNLL-2000 and LLL-2000*, pp. 142–144, 2000.

[8] Lance A. Ramshaw and Mitchell P. Marcus. Text chunking using transformation-based learning. In *Proceedings of the 3rd Workshop on Very Large Corpora*, pp. 88–94, 1995.

[9] Erik F. Tjong Kim Sang. Noun phrase recognition by system combination. In *Proceedings of ANLP-NAACL 2000*, pp. 50–55, 2000.

[10] Erik F. Tjong Kim Sang. Text Chunking by System Combination. In *Proceedings of CoNLL-2000 and LLL-2000*, pp. 151–153, 2000.

[11] Erik F. Tjong Kim Sang and Sabine Buchholz. Introduction to the CoNLL-2000 Shared Task: Chunking. In *Proceedings of CoNLL-2000 and LLL-2000*, pp. 127–132, 2000.

[12] Erik F. Tjong Kim Sang, Walter Daelemans, Hervé Déjean, Rob Koeling, Yuval Krymolowski, Vasin Punyakanok, and Dan Roth. Applying system combination to base noun phrase identification. In *Proceedings of COLING 2000*, pp. 857–863, 2000.

[13] Erik F. Tjong Kim Sang and Jorn Veenstra. Representing text chunks. In *Proceedings of EACL'99*, pp. 173–179, 1999.

[14] Manabu Sassano and Takehito Utsuro. Named Entity Chunking Techniques in Supervised Learning for Japanese Named Entity Recognition. In *Proceedings of COLING 2000*, pp. 705–711, 2000.

[15] Robert E. Schapir, Yoav Freund, Peter Barlett, and Wee Sun Lee. Boosting the marting: A new explanation for the effectivenss of voting methods. *The Annals of Statistics*, Vol. 26, No. 5, 1998.

[16] Bernhard Schölkopf, Chirs Burges, and Valdimir Vapnik. Incorporating Invariances in Support Vector Learning Machines. In *Artificial Neural Networks (ICANN)*, pp. 47–52, 1996.

[17] Hans van Halteren. Chunking with WPDV Models. In *Proceedings of CoNLL-2000 and LLL-2000*, pp. 154–156, 2000.

[18] Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

[19] 颯々野学, 宇津呂武仁. 統計的日本語固有表現抽出における固有表現まとめ上げ手法とその評価. 情報処理学会 自然言語処理研究回 NL139-2, pp. 1–8, 2000.

[20] 内元清貴, 青馬, 村田真樹, 小作浩美, 内山将夫, 井佐原均. 最大エントロピーモデルと書き換え規則に基づく固有名詞抽出. 自然言語処理, Vol. 7, No. 2, 2000.

[21] 平博順, 春野雅彦. Support Vector Machine によるテキスト分類における属性選択. 情報処理学会論文誌, Vol. 41, No. 4, p. 1113, 2000.