社団法人　情報処理学会　研究報告
IPSJ SIG Technical Report

2006－MPS－58（18）
2006／3／17

準最適解からの加重文脈自由文法の獲得

乾 伸雄, 品野 勇治 (東京農工大学大学院共生科学技術研究部)

本論文では多値分類 SVM を用いた文脈自由文法の学習方法について述べる。Tsochantaridis らは構文解析で得られた構文木を多値分類 SVM に与え，加重文脈自由文法 (WCFG) の重みを学習する方法を提案した。多値分類 SVM の記述能力は柔軟であり、WCFG を超えた精度が期待できる文法にも適用できる。我々はそのような文法の一つとしてアーク文脈自由文法 (ACFG) を提案する。しかしながら、ACFG で最適解を求める手間は WCFG のそれよりも大きい。そのため、WCFG と同じ手間で求められる ACFG の準最適解に適した重みの学習方法を提案する。数値実験で F 値を測定したところ、提案する準最適解を使った ACFG が 79.9 であるのに対し、PCFG が 76.6、WCFG が 77.6 であり、高い精度を達成できた。また、最適解を使った ACFG が 79.6 と同程度であったが、提案手法は短い時間で学習を終了することができた。

Acquiring Weighted Context-Free Grammars from Sub-optimal Solutions

Nobuo Inui, Yuji Shinano(Tokyo University of Agriculture and Technology)

We discuss a learning algorithm of context-free grammars. Tsochantaridis proposed a learning method for the weighted context-free grammar (WCFG), using multiclass SVM. Against this, we propose a multiclass SVM for the arc-weighted context-free grammar (ACFG) which represents inter-rule relations. For learning from sub-optimal syntactic trees of ACFG, we propose a new learning algorithm where partial syntactic trees are used. Experimental results showed that our method achieved 79.9 in F-value against 76.6 for PCFG and 77.6 for WCFG. Thought the precision of our method is almost same as ACFG with optimal solutions, the learning of our method completed within a shorter time.

1　はじめに

近年，サポートベクタマシン (SVM) による学習手法に注目が集まっている．SVM は二値分類を行う学習器として開発され，与えられた例に対する一般化能力が高いとされている．二値分類を多値分類に拡張した SVM(多値分類 SVM) も提案されている [2]．多値分類 SVM を用いて，加重文脈自由文法 (WCFG) における規則の重みを学習する研究 [1, 6] が行われた．WCFG では構文木はそれを構成する規則の重みの総和 (スコア) で評価され，最大のスコアを持つ構文木が構文解析器より出力される．

構文解析の目標の一つが解析精度の向上であることから，今日までに様々な方法が導入されてきた．その中の一つに，複数の規則を組み合わせて一つの規則として表現する Tree gram(T-gram) がある [5]．関連性の高い規則を組み合わせることで精度の高い構文解析が期待できる．我々は，加重文脈自由文法を拡張し，T-gram のような規則間の関連性を表現できるアーク加重文脈自由文法 (ACFG: Arc-weighted CFG) を提案する．ACFG に現れる重みを多値分類 SVM により学習することで精度の高い構文解析をめざす．

WCFG に比べ，ACFG の構文解析は原理的に時間がかかる．しかし，準最適解であれば，ACFG の構文解析は WCFG とほぼ同じ時間で行える．本論文では，この準最適解に適応した ACFG の重みを多値分類 SVM で学習する手法を提案する．

2　自然言語文法と構文解析

2.1　WCFG と ACFG

WCFG は $\{N, \Sigma, R, \Theta\}$ により定義される [1][1] ．N は非終端記号の集合，Σ は終端記号 (単語) の集合である．$R = \{r_1, \ldots, r_l\}$ は l を規則数とする規則の集合であり，規則は $A \rightarrow \alpha, A \in N, \alpha \in \{N \cup \Sigma\}^*$ の形式を持つ．本論文では，A を規則の左辺，α を規則の右辺とよぶ．Θ は個々の規則の重みの列ベクトルであり，$\Theta = (w_1 \ldots w_l)^t, w_j \geq 0, j = 1, \ldots, l$ により定義される．確率文脈自由文法 (PCFG) は，構文木の生成確率が確率の対数の和で表現されるため，WCFG の一種である．

単語列 x_i に対し，可能な構文木の集合を Y_i と記述する．$h_\Theta(x_i) = argmax_{y \in Y_i} \phi(x_i, y)\Theta$ は，x_i に対する最尤な構文木を与える．ここで，$\phi(x_i, y) = (c_1 \ldots c_l), c_j \in Z, j = 1, \ldots, l$($Z$ は 0 以上の整数の集合) は単語列 x_i に対する構文木 y における規則の出現頻度を表すベクトルである．この規則の頻度と重みの積和をスコアとよぶ．

次に，CFG に構造を表す T-gram の要素を取り入れた ACFG を $\{N, \Sigma, R, \Pi\}$ により提案する．N，Σ，R

[1] 正確には開始記号が文法に与えられるが，本論文では省略する

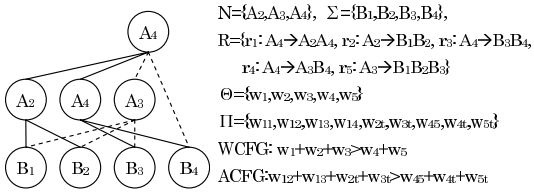

図 1: WCFG および ACFG の例
(実線が正例，点線が負例，重みの添え字は規則の番号あるいは終端記号を意味する．“t” は終端記号を表わす．例えば，右辺に終端記号がある規則は,w_{2t} など終端記号に関する重みが付与される．WCFG および ACFG それぞれについて，不等式を満す場合，構文解析の結果正例が得られる．)

左辺　　右辺
接尾語句 → 接尾語句 記号 接尾語句 記号 接尾語句
動詞句　 → 助詞句 動詞句
名詞句　 → 名詞 記号 名詞 名詞句

図 2: EDR コーパスより抽出された規則の例

は WCFG と同じである．Π は規則間の重みを表す列ベクトルであり，$\Pi = (w_{11} \ldots w_{1l} \ldots w_{l1} \ldots w_{ll})^t, w_{jk} \geq 0, j, k = 1, \ldots, l$ によって与える．w_{jk} は，r_j の右辺の非終端記号に r_k の左辺が対応する重みである．最適な構文木は $h_\Pi(x_i) = argmax_{y \in Y_i} \psi(x_i, y)\Pi$ によって与える．$\psi(x_i, y) = (c_{11} \ldots c_{1l} \ldots c_{l1} \ldots c_{ll}), c_{jk} \in Z, j, k = 1, \ldots, l$ である．c_{jk} は，規則 r_j の直下に r_k が出現した頻度を表す．図 1 に，WCFG と ACFG の例を示す．この例が示すように，スコアを計算する重みの数は，WCFG では構文木を構成する規則数，ACFG ではアーク数に等しい．

ACFG は WCFG を一般化したものである．これは，ある規則の右辺に現れる非終端記号を左辺に持つ全ての規則に対して，同じ重みが割り振られる場合，ACFG は WCFG と同等であることより示される．そうでない場合は，ACFG では生成可能な構文木が WCFG で生成できない場合が存在し，分類能力 (正しい構文木を生成する能力) は ACFG の方が高い．

2.2　ボトムアップ構文解析

構文解析は単語列を構文木に変換するタスクである．WCFG は個々の規則が独立に重みを持つので，部分単語列に対して，非終端記号ごとに最尤な構文木を求めていくことで構文解析が行える．例えば，図 1 で，部分単語列 $B_1B_2, B_1B_2B_3, B_3B_4$ それぞれに対して最尤な部分構文木が求まったとする[2]．$B_1B_2B_3B_4$ に対する最尤な構文木のスコアはこれらの部分構文木のスコアを使うことで求められる．このように，WCFG の構文解析は，

[2] 非終端記号ごとに求める

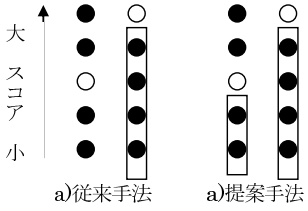

図 3: 経験損失
(白丸が正しいクラス，黒丸が誤ったクラス，四角で囲まれたクラスは学習に寄与している誤ったクラス)

短い部分単語列から順番に最尤な部分構文木を決定するというボトムアップ構文解析により実現できる[3]．

これに対し，ACFG は重みが規則間で決定されるため，規則ごとに部分構文木のスコアを保持する必要がある．一般に非終端記号の数に比べ規則の数は多いため，構文解析にかかる計算時間は WCFG よりも ACFG の方が長い．

本論文では，準最適解を「最適解と同じスコアになりえるがその保障がない解」と定義する．ACFG の準最適解は，WCFG の構文解析方法と同様に実現できる．すなわち，部分単語列に対して非終端記号ごとに最尤な部分構文木を求め，その最尤な部分構文木の根で適用された規則とそれらをまとめる規則間の重みを求める．ボトムアップに最適解を構成する部分構文木と同じ部分構文木が選ばれている場合，準最適解は最適解となる．

さらに，構文解析の実行時間を短縮するため，参考文献 [4] を参考にし，次のような形式の規則を使う．これによって，部分単語列より一意に非終端記号が決定されることになる．実際の文法の例を図 2 に示す．

$$A \rightarrow BA|Ba, a \in \Sigma, A = \delta(a), B = \{N \cup \Sigma\}^*$$
$$where \ \ \delta(a) : \Sigma \cup N \rightarrow N$$

3　多値分類 SVM による学習

3.1　経験損失

多値分類 SVM の定式化では，最小化される経験損失を定義する．経験損失は正例のスコアが最大のスコアを持つ負例のスコアよりも高くなるように設定される [2] (従来手法)．これは負例のスコアが正例のスコアよりも大きい場合，その例に関する負例全ては学習に寄与しないという問題がある．

これに対し，我々は**正例のスコアより大きいスコアを持つ負例の数を最小化**する経験損失を提案する (提案手法)．この定義では正例のスコアより小さいスコアを持

[3] チョムスキー標準形に対する CYK 法を拡張したものである

$$\mathrm{Model}(Y,\Pi):$$
$$\mathrm{Min}\ \frac{C}{|I|}\sum_{(x_i,y_i)\in I}\frac{\sum_{y_j[p,q]\in PTS(x_i,y_i)}\xi_{ij}[p,q]}{|PTS(x_i,y_i)|}+\frac{1}{2}||\Pi||^2$$
$$(\psi(x_i,y_i[p,q])-\psi(x_i,y_j[p,q]))\Pi\geq 1-\xi_{ij}[p,q]$$
$$\forall(x_i,y_i)\in I,\forall y_j\in Y_i-\{y_i\}\subset Y,\forall y_j[p,q]\in PTS(x_i,y_i) \quad (1)$$
$$0\leq\xi_{ij}[p,q]\ \ \forall(x_i,y_i)\in I,\forall y_j\in Y_i-\{y_i\},\forall y_j[p,q]\in PTS(x_i,y_i) \quad (2)$$
$$where PTS(x_i,y_i)=\{y_j[p,q]|y_j\in Y_i-\{y_i\},1\leq p<q\leq \mathrm{length}(x_i),$$
$$y_i[p,q]\neq\mathbf{0},y_j[p,q]\neq\mathbf{0},y_i[p,q]\neq y_j[p,q]\}$$
$y_i[p,q]$ は部分単語列 $s_p\ldots s_q$に対応する部分構文木であり，存在しない場合は $\mathbf{0}$ となる

図 4: 多値分類 SVM による ACFG に対する重み決定の定式化

つ負例は全て学習に貢献し，学習に寄与しない例の数を少なくできる．提案手法と従来手法の関係を図 3 に示す．

3.2 多値分類 SVM の定式化

ACFG において，準最適解を求める構文解析はボトムアップに準最適解となる部分構文木を構成することで行われることを述べた．このため，従来手法 [6] による構文木全体で正例と負例の差分を取る方法を用いると，学習結果と構文解析結果にギャップが現れる．この問題を，本論文では**部分構文木間の差分**を取る多値分類 SVM により解決する．定式化を図 4 に示す．

Y は負例の全集合である．ξ_{ij} は SVM のソフトマージンを実現するためのスラック変数である．ξ_{ij} は目的関数において最小化され，図 4 式 (1) において，1 未満の場合，正例 y_i と負例の y_j は正しく判別され，1 以上の場合は判別されないことを表す．目的関数の C は正の定数であり，経験的に決められる．C の値が大きければ，訓練データにフィットするよう重みが決定され，小さければ，一般的な重みが決定される．$\mathrm{length}(x_i)$ は単語列 x_i の長さを表し，単語列 $s_1\ldots s_n$ に対して，整数 p，q は $1\leq p<q\leq n$ となる．$y_i[p,q]$，$y_i[p,q]$ は単語列 $s_p\ldots s_q$ をカバーする構文木である．正例あるいは負例において，部分構文木を形成しない部分単語列は考慮しない．$PTS(x_i,y_i)$(Partial Tree Set) は，正例 y_i の部分構文木と一致しない部分構文木を要素に持つ集合である．このように部分構文木を用いることで準最適解に対する重みが学習される．

与えられた単語列に対し，生成可能な構文木の集合は一般的に巨大であり，全てを列挙することは困難である．このため，従来手法 [6] では構文解析によって得られた構文木を集合 Y_i に加える学習方法を提案した．新たな構文木が集合に加えられなくなるまで学習は行われる．我々もこの従来手法と同様の方法で学習を行う．ただし，

表 1: 実験で用いた EDR コーパス中のデータセット

	訓練データ				評価データ
名称	t1	t2	t3	t4	e
文数	374	724	1,798	3,592	3,543
総規則数	6,171	12,208	30,768	61,653	60,766
異なり規則数	362	438	663	915	926
(e と共通)	298	340	453	537	-

表 2: 大規模な実験で用いたデータセット

名称	e1		e2	
用途	訓練	評価	訓練	評価
文数	3,592	18,180	3,814	17,958
異なり規則数	915	2,406	1,132	2,270
共通の異なり規則数	748		829	

多値分類 SVM で部分構文木を扱うので，集合 Y_i の要素は部分構文木となる．部分構文木は，構文解析がボトムアップに行われるのに従い，**部分単語列の短い部分構文木から順番**に選ぶ．ただし，一回の構文解析で抽出される部分構文木は一本とする．

4 数値実験

数値実験は，PCFG，WCFG，ACFG を対象に行った．WCFG に対しては，従来手法の構文木全体でスコアを比較する手法 (W) と提案手法の部分構文木のスコアを比較する手法 (S) を試みた．ACFG に対しては，構文解析器が準最適解を出力する場合の W と S および最適解を出力する場合の W(W(opt)) を試みた．実験環境は，Pentium4 3.4GHz, メモリ 1GB の PC 上で，構文解析は cygwin の C 言語で記述した．多値分類 SVM を解く 2 次計画問題ソルバに ILOG CPLEX 9.1 を用いた．

学習対象は EDR コーパス [3] から抽出した．このデータを表 1，表 2 に示す．本実験では学習手法の評価に着目するため，構文規則は訓練データおよび評価データの両方から抽出した．

表 3: 実験結果 (評価データ)

文法	アルゴリズム	PCFG	WCFG		ACFG		
			W	S	W	S	W(opt)
t1	R(%)	67.5	77.6	77.6	78.2	78.8	78.2
	P(%)	76.2	75.1	75.0	75.6	76.2	76.0
	F	71.6	76.3	76.3	76.8	77.4	77.1
	It(回)	-	81	131	85	50	73
	Ti(秒)	-	4,196	6,757	5,123	2,911	4,783
	PT(%)	-	97.8	97.7	84.1	86.8	81.8
t2	R(%)	72.8	78.0	78.0	79.0	79.8	79.2
	P(%)	78.0	75.6	75.6	76.6	77.4	77.0
	F	75.3	76.8	76.8	77.7	78.6	78.1
	It(回)	-	83	149	97	78	129
	Ti(秒)	-	8,332	14,929	11,876	9,132	16,217
	PT(%)	-	97.8	97.6	80.4	83.3	83.2
t3	R(%)	74.5	78.3	78.4	80.0	80.6	79.8
	P(%)	78.0	76.0	76.2	77.7	78.3	77.7
	F	76.2	77.2	77.3	78.8	79.4	78.7
	It(回)	-	83	113	94	67	87
	Ti(秒)	-	21,665	29,324	31,668	21,683	30,438
	PT(%)	-	97.1	97.3	75.8	78.4	75.9
t4	R(%)	75.2	79.0	78.8	80.4	81.2	80.8
	P(%)	78.1	76.3	76.2	77.7	78.5	78.5
	F	76.6	77.6	77.5	79.0	79.9	79.6
	It(回)	-	98	91	103	107	119
	Ti(秒)	-	51,892	47,890	74,155	58,960	75,968
	PT(%)	-	96.3	96.5	70.6	75.9	71.6

表 4: 大規模な実験の結果 (評価データ)

文法	アルゴリズム	PCFG	WCFG	ACFG	
			W	S	W(opt)
e1	R(%)	69.9	76.3	79.4	78.0
	P(%)	76.0	73.9	74.8	75.8
	F	72.8	75.1	77.0	76.9
	It(回)	-	83	94	153
	Ti(秒)	-	94,577	119,868	202,586
	PT(%)	-	98.0	87.8	86.9
e2	R(%)	69.1	77.7	80.3	78.3
	P(%)	74.6	74.7	78.1	76.3
	F	71.7	76.2	79.2	77.3
	It(回)	-	125	90	136
	Ti(秒)	-	94,577	118,189	203,417
	PT(%)	-	97.4	85.5	78.8

表 3, 表 4 に精度および学習時間を示す．ここで，R は再現率，P は適合率，F は F 値を示す．再現率は正例に含まれる規則数に占める正解となった規則数，適合率は構文解析結果に含まれる規則数に占める正解となった規則数を表わす．F 値は $\frac{2RP}{R+P}$ で計算される．It は学習が収束するまでの多値分類 SVM の適用回数，Ti は学習にかかった総時間，PT は構文解析が占める時間の割合を示す．

表 3 は，訓練データのサイズの違いに対する結果を示す．PCFG に比べ，多値分類 SVM で学習した WCFG，ACFG では安定した精度が得られる．WCFG において提案手法の多値分類 SVM は従来手法と同程度の精度を示した．部分構文木を用いると，局所的な収束を原因とする過学習の恐れがあるが，その傾向はみられなかった．しかし，t4 を除いて，学習時間は長かった．多値分類 SVM の適用回数が多かったことが原因であり，最適解が求められる WCFG については提案手法は適切ではないと言える．

表 4 には，評価データを大きくし，規則数を増やしたものに対する実験結果を示した．表 3，表 4 より，提案手法である ACFG の解析精度は，PCFG および WCFG を有意に上回るものであり，文法モデルとして有効であることがわかった．ACFG は重みの数が多いことから，同じ訓練データサイズの WCFG よりも過学習となる恐れがあるが，その傾向はみられなかった．ACFG で S を用いた場合，W よりも精度，学習時間の点で優れていた．これは，準最適解を求める構文解析と学習アルゴリズムが適合していることを示している．W(opt) に対して，提案手法の精度は同程度であった．これは，準最適解から重みを学習する手法でも最適解から学習する手法と同じ質の文法が得られることを示している．学習時間に関しては，提案手法は W(opt) よりも優れていた．

これらの結果より，提案手法である ACFG および準最適解を使った多値分類 SVM による学習手法は，従来研究より優れていると考えられる．

5 おわりに

本論文では，自然言語構文解析を例に取り，準最適解からの多値分類 SVM を使った学習のアルゴリズム，正例よりスコアの高い負例の数を最小化する経験損失を提案した．そして，ACFG を提案し，数値実験によりその有効性を検証した．

多値分類 SVM は柔軟な記述性を持ち，本論文の提案手法によれば，最適解が得られなくても学習が成功できるので，今後，より複雑な対象に対して研究を行いたいと考えている．

参考文献

[1] M. Collins: Parameter Estimation for Statistical Parsing Models: Theory and Practice of Distribution-Free Methods, New Development in Parsing Technology, Kluwer Academic, 2004

[2] K. Crammer, Y. Singer: On the Alogorithmic Implementation of Multiclass Kernel-based Vector Machines, J. of Machine Learning Research, Vol.2, pp.265-292, 2001

[3] EDR:EDR Electric Dictionary Manual Ver. 1.5, 1996

[4] 白井清昭，徳永建伸，田中穂積 : EDR コーパスからの確率文脈自由文法の自動抽出に関する研究，EDR 電子化辞書利用シンポジウム， pp.57-62，1995

[5] K. Sima'an: Tree-gram Parsing Lexical Dependencies and Structural Relations, Proc.of the 38th Annual Meeting on ACL, pp.37-44, 2000

[6] I. Tsochantaridis, T. Hofmann, T. Joachims, T. Altun: Support Vector Machine Learning, for Interdependent and Structure Output Space, International Conference on Machine Learning, pp.823-830, 2004