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

2005—NL—167（4）
2005—SLP— 56（4）
2005／5／26

Stackingの効率的な学習方法と日本語固有表現抽出での評価

岩倉 友哉
株式会社富士通研究所
神奈川県川崎市中原区上小田中 4-1-1
iwakura.tomoya@jp.fujitsu.com

概要：本論文では，分類器の出力を混合する手法の一つである Stacking を効率的に学習するための手法を提案する．Stacking 学習では，混合に利用する分類器作成用の学習データと，Stacking 用学習データの二種類が必要となる．本手法では，交差検定の考えを応用して，分類器作成とテストを繰り返すことで，分類器作成のための学習データから Stacking 学習用の学習データを獲得する．CRL 固有表現データを学習データとし，IREX 一般課題で提案手法を評価した．本手法は，より細かい単位で交差検定を行なって Stacking 用学習データを獲得することで精度を改善し，F 値で 85.83 という従来手法より良い結果が得られることを確認した．

キーワード：Stacking，日本語固有表現抽出，機械学習

Efficient Stacking Learning Method and its Evaluation with Japanese Named Entity Extraction

Tomoya Iwakura
Fujitsu Laboratories Ltd.
1-1, Kamikodanaka 4-chome, Nakahara-ku, Kawasaki 211-8588.

Abstract：In this paper, we propose an efficient stacking learning algorithm. For stacking learning, a training data for a stacking model construction is required apart from a training data for each model construction to be stacked. Our stacking learning algorithm acquires a training data for stacking by conducting cross-validation to a training data for each model construction. The cross-validation results are used as a training data for stacking. We evaluate our stacking method with IREX formal test data called GENERAL for evaluation and 'CRL data' for training. Our method improves accuracy to acquire a training data for stacking by conducting cross-validation with a larger number of subsets and attains F-measure of 85.83. This result is better than previous one in this literature under the same condition.

Keywords：Stacking，Japanese Named Entity Extraction, Machine Learning

1　はじめに

近年，コーパスベースの言語処理研究が広く行なわれている．コーパスベースの言語処理では，正解タグが付与されたコーパスである学習データから機械学習アルゴリズムを使って規則獲得やモデル構築を行なうことで自然言語処理ツールを開発でき，形態素解析や係り受け解析，固有表現抽出などの様々な自然言語処理タスクに適用されている．

機械学習アルゴリズムには，Support Vector Machines (SVMs)[8] や Boosting[6] などがある．コーパスベースの言語処理では，利用する機械学習アルゴリズムを替えることで，同一の学習データの同一素性集合から異なる規則やモデルを持つ分類器を作成できる．異なる機械学習アルゴリズムから作成される分類器ではそれぞれの利点・欠点が異なる．それぞれの利点・欠点をうまく利用できる分類器を作成することができれば，より良い結果を得られると期待できる．そのための手法として，Voting と Stacking がある．

Voting とは，各分類器による分類結果をスコア付けし，ある閾値を用いて，最終的な結果を得る手法である．Florian ら [3] は，CoNLL2003 の Shared Task である固有表現抽出において，Voting によって単独の分類器よりも良い結果を残している[1]．

[1] 各モデルが出力する結果を手掛りとして追加する Stacking 的な実験結果も報告しているが，実験方法の詳細は述べられていない．

Votingは，それぞれの分類器が出力した分類結果を数値化して，最終的な分類を決定する．そのため，各事例に対するそれぞれの分類器の振舞いを細かく考慮することを考えた場合，より複雑なスコア付け手法が必要となる．

Stackingとは，複数分類器の事例に対する分類結果を素性として利用することで分類を行なう手法である．Utsuroら [7] は，3種類の日本語固有表現抽出器の出力をStackingする実験を行なっており，それぞれの抽出器を単独で利用するより良い結果が残せると報告している．

Stackingでは混合に利用される分類器による分類結果を素性として持つ学習データから学習を行なう．そのため，Votingより比較的簡単に各事例に対するそれぞれの分類器の振舞いを学習できると言える．しかし，Stacking学習では1段目で利用する分類器の構築用とStacking学習用の2種類の学習データが必要であり，効率的にStackingを学習するためにはStacking用の学習データを低コストで獲得することが必要である．

本論文では，人手で新たに学習データ用意することなく，Stacking用の学習データを獲得するStacking学習手法を提案する．2節で提案するStacking学習手法について説明する．3節で，Stacking手法の評価を行なう日本語固有表現抽出について説明し，4節について実験結果を述べる．

2 Stacking学習手法

Stackingの実現には，Stackingの1段目に利用する分類器作成用の教師データと，Stacking用の学習データの2種類が必要となる．Stacking用の学習データは，手掛りに利用するそれぞれの分類器による分類結果である分類結果素性を持つ必要がある．Stacking用の学習データ獲得方法としては，たとえば，次の3つが考えられる．

- 手法 (a): 1段目の分類器作成に利用する学習データからStacking用の学習データを作成する．この手法では，1段目用の学習データのそれぞれの事例に付与されている正解タグを，それぞれの事例の分類結果素性とすることでStacking用の学習データを作成する．たとえば，ある事例の正解がPERSONである場合，その事例の分類結果素性には全てPERSONが追加される．この方法で作成される学習データでは，事例の正解タグとそれぞれの分類結果素性は常に同じとなる．

- 手法 (b): 手法 (a) と同様に1段目の分類器作成に利用する学習データからStacking用の学習データを作成する．この手法では，1段目用の学習データから分類器を作成し，その分類器で1段目用の学習データを分類する．この分類結果を1段目用の学習データに分類結果素性として追加してStacking用の学習データを作成する．

- 手法 (c): 1段目で利用する分類器の学習データとStacking用の学習データを別々に用意する．1段目で利用する分類器を作成したあと，Stacking用の学習データを分類し，その分類結果を分類結果素性としてStacking用の学習データに追加して作成する．

```
STEP 1: 使用する学習器を用意.
STEP 2: 学習データを N 分割.
STEP 3: STEP 1 で用意した全ての
        学習器で以下の処理を実施.
 for i := 1 to N
 begin
  i 番目以外のデータ (N-1 個) から分類器を作成.
  作成した分類器を使い i 番目のデータでテスト.
  i 番目のデータに分類結果を素性として加えたものを
    Stacking 用の学習データとして保持.
 end
STEP 4: STEP 3 で得られた
        Stacking 用学習データから Stacking を学習.
```

図 1: Stacking学習アルゴリズム

Feature	分類器 1- 分類結果素性	分類器 2- 分類結果素性	tag
宮崎 名詞	S-PER	S-LOC	S-PER
さん 名詞	O	O	O
は 助詞	O	O	O
宮崎 名詞	S-PER	S-LOC	S-LOC
の 助詞	O	O	O
出身 名詞	O	O	O
. 句点	O	O	O

図 2: 獲得されるStacking用学習データの例

しかし，手法 (a)〜(c) には次のような欠点があると考えられる．

- 手法 (a) では，簡単にStacking用の学習データを作成できる．しかし，分類結果素性は正解タグと同じであるため，それぞれの分類器の振舞いを学習できない．

- 手法 (b) は，SVMsのように学習データに対してはほぼ完璧に分類できる分類器を作成可能な学習器を利用した場合，手法 (a) とほぼ同じ結果になる恐れがある．

- 手法 (c) は，Stacking用学習データを別に用意し，そのテスト結果を使うので，それぞれの分類器の振舞いを学習することができる．しかし，学習データ作成の負荷が問題となる．また，分類器作成用の学習データとStacking用の学習データのどのような比率で用意するのが適切であるかの判断が難しい．

Stackingを効率的に学習するためには，単一の学習データから手法 (c) で獲得されるようなStacking用学

習データを単一の学習データから獲得できれば良い．Stacking 用の学習データを 1 段目の分類器構築用の学習データとは別に自動的に獲得することはほぼ不可能である．しかし，学習データの一部を削除することで全てを学習したときの近似の分類器は作成でき，その近似の分類器で残りの学習データを分類することで，全体の学習データを利用した場合に近い Stacking 用の学習データを獲得することができると考えられる．そこで，交差検定の考えを応用した Stacking 用学習データ獲得方法を提案する．

図 1 がそのアルゴリズムである．STEP 1 では，SVMs や Boosting など分類器作成に使用する学習器を用意する．続いて，STEP 2 にて，学習データを分割する．全てを学習した場合に近い結果を得るためには，細かく分割する必要がある．STEP 3 では，STEP 1 で用意したそれぞれの学習器を使って，学習：テスト =(N-1)：1 の交差検定を行ない，テスト結果を Stacking 用の学習データとする．日本語固有表現抽出であれば，図 2 のような，それぞれの分類器による分類結果を分類結果素性として持つ学習データを獲得できる．STEP 4 では，STEP 3 で得られた学習データから，Stacking の学習を行なう．たとえば，STEP 3 で分類器作成に利用した素性に加え，それぞれの学習器から作成されたモデルの出力結果を素性として追加して学習する．

本手法は，学習データを細かい単位に分割して，交差検定を行なうことで，全てを学習したときに作成される分類器を近似でき，単一の学習データから効率的に質の良い Stacking 学習データを獲得できると期待される．

3 日本語固有表現抽出

2 節で述べた Stacking 手法の評価を行なう日本語固有表現抽出タスクと Stacking に利用する日本語固有表現抽出器の実装方法について説明する．固有表現抽出とは，テキストから “人名” や “地名” などの固有名詞のクラスや “日付” や “時間” などの数値表現のクラスを抽出するタスクである．これ以降では，固有表現を Named Entity あるいは NE と記す．表 1 は，IREX[1] で定義された 8 種類の NE クラスの例である．日本語の NE 抽出は形態素解析器に区切られた単語単位で処理を行なうのが一般的であるが，単語の区切りと NE の区切りが必ずしも一致しないという問題がある．そのため，単語を NE クラスを判別する NE Tagging に加えて，単語の一部が NE である場合に対処するための後処理が必要となる．それぞれの処理について 3.1 節と 3.2 節で説明する．

3.1 NE Tagging

NE Tagging とは，形態素解析器の出力した単語がどの NE クラスになるかを判別する処理である．NE Tagging の実装は，調査範囲で IREX 総合課題に対し最も良い結果であった磯崎ら [10] の手法を参考に実装した．今回の NE Tagging 実装では，形態素解析器は ChaSen[2]を利用した．NE Tagging で使用した素性は次の 3 種類である．

- **表記**: ChaSen によって区切られる単語の表記．
- **品詞**: ChaSen によって単語に付与された品詞．ただし，未知語と判別された場合は “固有名詞-一般” として出力するように ChaSen の設定を変更した．
- **文字種**: 平仮名，片仮名，漢字，アルファベット大文字，アルファベット小文字，記号・その他，の 6 種類の組み合わせを文字種とした．同じ文字が 2 回以上続く場合は，“+” を付けて区別した．たとえば，「訪朝」であれば，漢字が二文字続くので，「漢字+」，「食べる」であれば，「漢字-平仮名+」という形で表した．数字は，漢数字，アラビア数字ともに正規化を行い，“12 以下”，“24 以下”，“100 以下”，“2000 以下”，“2000 より大きい”，という分類を行なった結果を文字種とした．

NE は複数の形態素から構成される場合があるため，NE タグは「形態素の “NE のクラス” と “NE 内での位置”」にて表現することにする．“⟨ORGANIZATION⟩ 岩倉使節団 ⟨/ORGANIZATION⟩ は訪 ⟨LOCATION⟩ 米 ⟨/LOCATION⟩ した” という文は，

岩倉 “BEGIN-ORGANIZATION”
使節 “INSIDE-ORGANIZATION”
団 “END-ORGANIZATION”
は “O”
訪 “O”
米 “SINGLE-LOCATION”
し “O”
た “O”

と表現される．今回の判別対象となる NE タグは，“NE のクラス” は表 1 にある 8 種類と，“NE 内での位置” である単独 (S-) 先頭 (B-)，中間 (I-)，終わり (E-) の 4 種類の組み合わせ 32 種類と，NE 以外 (O) の 1 種類，合計 33 種類である．

NE Tagging の手がかりに利用する形態素範囲は，対象の形態素および前後 2 形態素，合計 5 形態素とした．それぞれの NE タグの NE クラスへのまとめあげは，Viterbi アルゴリズムにて行なう．Viterbi アルゴリズムへの入力は確率値である．しかし，今回利用する学習器から作成される分類器は確信度を返すため，それぞれの確信度を sigmoid 関数にて正規化を行い確率値の代わりとした[3]．

[2] http://chasen.naist.jp/hiki/ChaSen/

[3] s(X) = 1 / (1 + exp(-βX)) の式で正規化する．この実験では，β は 5 とした．

表 1: IREX で定義された Named Entity の例

NE	ARTI -FACT	DATE	LOCA -TION	MONEY	ORGANIZA -TION	PERCENT	PERSON	TIME
例	ノーベル化学賞	5月5日	日本	２００円	富士通	3割	太郎	午前 10:00-

単語	品詞	文字種	NEtag
岩倉	名詞-固有名詞-組織	漢字+	B-ORGANIZATION
使節	名詞-一般	漢字+	I-ORGANIZATION
団	名詞-接尾-一般	漢字	E-ORGANIZATION
は	助詞-係助詞	平仮名	O
訪米	名詞-サ変接続	漢字+	S-LOCATION
し	動詞-自立	平仮名	O
た	助動詞	平仮名	O

→

文字	品詞	文字種	単語内での位置+NEtag	NEtag
岩	B-名詞-固有名詞-組織	漢字	B-B-ORGANIZATION	B-ORGANIZATION
倉	E-名詞-固有名詞-組織	漢字	E-B-ORGANIZATION	I-ORGANIZATION
使	B-名詞-一般	漢字	B-I-ORGANIZATION	I-ORGANIZATION
節	E-名詞-一般	漢字	E-I-ORGANIZATION	I-ORGANIZATION
団	S-名詞-接尾-一般	漢字	S-E-ORGANIZATION	I-ORGANIZATION
は	S-助詞-係助詞	平仮名	S-O	O
訪	B-名詞-サ変接続	漢字	B-S-LOCATION	O
米	E-名詞-サ変接続	漢字	E-S-LOCATION	B-LOCATION
し	S-動詞-自立	平仮名	S-O	O
た	S-助動詞	平仮名	S-O	O

図 3: 素性表現の例: 左側は NE Tagging，右側は後処理の素性表現

3.2 後処理

後処理とは，単語の一部が NE である場合に，単語から切り出しを行なう処理である.「岩倉 / 使節 / 団 / が / 訪米 / し / た」と形態素解析した文から NE 抽出を行なう場合,「岩倉使節団」は ORGANIZATION として抽出できるが，LOCATION である「米」を単語「訪米」から抽出することができない.

この問題に対し，Asahara ら [2] は文字単位での NE 抽出を提案している．今回の実験では，単語単位での NE Tagging を行なったあとに文字単位の NE Tagging を後処理として適用する.

図 3 は単語単位から文字単位への変換例であり，図 3 の左側の NE Tagging 結果が，図 3 の右側に変換される．後処理で判別対象となる NE タグは，表 1 にある 8 種類の “NE のクラス” と “NE 内での位置” である先頭 (B-)，中間 (I-) の 2 種類の組み合わせ 16 種類と，NE 以外 (O) の 1 種類，合計 17 種類である．後処理で利用する素性は，文字，文字が属する単語の品詞と位置，文字種，NE Tagging 結果である．それぞれの素性は，文字の単語内での出現位置である，先頭 (B-)，中 (I-)，終わり (E-)，単独 (S-) の 4 つを付与したものを利用した．手がかりに利用する文字の範囲は，現在の位置に加え，前後 2 文字を利用する．解析は文末から文頭の方向へ行い，2 つ前までに決定した NE タグを動的に素性として利用する.

4 実験・評価

本節では提案する Stacking 手法および手法 (a)〜(c) の日本語 NE 抽出による評価結果について述べる．まず，4.1 節にて，Stacking に利用する機械学習アルゴリズムについて説明し，4.2 節にて，Stacking の評価結果について述べる.

4.1 機械学習アルゴリズム

Stacking 手法の評価に使用する NE 抽出器の実装に，次の 3 つの機械学習アルゴリズムを用いた.

- Support Vector Machines (SVMs): SVMs[8] に基づくチャンカである YamCha[4] を利用する．今回の実験では，C の値は 1，polinominal kernel 2 次とし，マルチクラスは，One Vs the Rest での学習とする.

- Boosting：今回の実験では，BoosTexter[6] の Real-valued prediction を弱学習器とする手法を，マルチクラスを One Vs the Rest で学習させるように変更したものを利用する．今回の実験では，Boosting の繰り返しは全てのクラスともに 1000 回に固定する.

- ADTrees：decision stump をノードとする分類木を boosting の枠組で作成する ADTrees アルゴリズム [4] を改良したもの利用する．ADTrees では，新規に選択された素性 (ノード) を親とし，“選択された素性が出現した事例”，“選択された素性が出現しなかった事例” の 2 つが新たなノード探索の候補として追加される．しかし，NE 抽出においては，素性数や事例数が多いため学習が進むにつれ，探索空間が急激に拡大する．今回の実験では，探索空間の追加は，“選択された素性が出現した事例” に限定することにする．また，ノード選択条件を，BoosTexter の “real-valuedpredicions and abstaining” に変更する．マルチクラスは，One Vs the Rest での学習とし，今回の実験では，Boosting の繰り返しは全てのクラスともに 1000 回に固定し，木の最大深さは 2 と設定した.

NE Tagging の学習には，SVMs，Boosting，ADTrees を利用し，後処理と Stacking の学習には，SVMs を利用する．今回の実験では，NE Tagging の部分で Stacking を行なう．Stacking とは関係がない後処理については，いづれの実験においても SVMs から作成されたものを利用することにした.

[4] http://chasen.org/~taku/software/yamcha/

表 2: SVMs による NE 抽出の評価結果

NE	Recall	Precision	F-measure
ARTIFACT	36.73	42.86	39.56
DATE	93.50	95.57	94.53
LOCATION	81.01	87.99	84.36
MONEY	100.00	100.00	100.00
ORGANIZATION	74.81	83.86	79.08
PERCENT	95.24	100.00	97.56
PERSON	88.45	88.70	88.58
TIME	94.92	98.25	96.55
All	82.86	87.98	85.34

表 3: Boosting による NE 抽出の評価結果

NE	Recall	Precision	F-measure
ARTIFACT	28.57	35.00	31.46
DATE	89.53	95.75	92.54
LOCATION	80.53	85.24	82.82
MONEY	100.00	100.00	100.00
ORGANIZATION	68.12	82.81	74.75
PERCENT	90.48	95.00	92.68
PERSON	82.82	87.76	85.22
TIME	86.44	94.44	90.27
All	78.49	86.42	82.27

4.2 Stacking 手法の評価

今回の実験では，2 節で述べた Stacking 手法を使って CRL 固有表現データから作成される NE 抽出器による IREX 一般課題 (GENERAL) の抽出結果で比較を行なう[5]．CRL 固有表現データとは，毎日新聞 95 年度版 1,174 記事，約 11,000 文に対して IREX で定義された固有表現を付与したデータである．

まず，3.1 節で紹介した 3 つの機械学習アルゴリズムで，Stacking に利用する 1 段目の NE 抽出器を作成し，GENERAL で評価した．評価には Recall，Precision，F-measure (F 値) を用いた．計算は次の式で行なう．

Recall = NUM/(抽出すべき NE 数)

Precision = NUM/(NE 抽出器が抽出した NE 数)

F-measure = 2×Recall×Precision/(Recall+Precision)

NUM とは，NE 抽出器が正しく抽出した NE 数である．

SVMs による抽出器での結果が表 2，Boosting による抽出器での結果が表 3，ADTrees による抽出器での結果が表 4 である．単独の機械学習アルゴルズムの中では SVMs による抽出器の結果が F 値で 85.83 と最も良く，ADTrees による抽出器では F 値で 83.43，Boosting による抽出器では F 値で 82.27 となった．今回の実験では表 2 にある SVMs による抽出結果の F 値をベースラインとする．

続いて，2 節で述べた手法 (a)〜(c) および今回提案する Stacking 手法について評価を行なう．Stacking 学習において利用する学習器の組み合わせは，SVM-Boosting，SVM-ADTrees，Boosting-ADTrees，SVM-Boosting-ADTrees の 4 種類である．Stacking の学習では，NE Tagging で使用する素性と，SVMs，Boosting，ADTrees から作成された NE 抽出器の出力結果を素性として利用する．表 5〜表 12 がそれぞれの Stacking 手法の評価結果である．vsSVM とはベースラインである表 2 の結果との F 値の差であり，太字はベースラインより上昇した項目である．

表 4: ADTrees による NE 抽出の評価結果

NE	Recall	Precision	F-measure
ARTIFACT	24.49	46.15	32.00
DATE	88.81	92.83	90.77
LOCATION	81.01	84.67	82.80
MONEY	100.00	100.00	100.00
ORGANIZATION	69.92	83.95	76.30
PERCENT	90.48	95.00	92.68
PERSON	86.20	89.21	87.68
TIME	96.61	98.28	97.44
All	79.95	87.23	83.43

手法 (a) の評価結果が表 5 である．全ての組み合わせにおいて，全体の F 値はベースラインである表 2 を下回る結果となった．個々の NE の F 値も，Bsooting-ADTrees の TIME と SVM-Boosting-ADTrees の TIME と DATE 以外は，ベースラインを下回っている．この結果は，2 節で述べた個々の分類器の振舞いを学習できないという手法 (a) の欠点を示す結果である．

手法 (b) の評価結果が表 6 である．この結果のうち，SVMs を含む組み合わせは，ベースラインである SVMs による結果と同じになってしまった．この結果が，2 節で述べた手法 (b) の欠点によるかを調査するために，それぞれの学習器が CRL データから作成したモデルを使って，CRL データの分類精度を調べたところ，SVMs による結果は F 値で 99.4，Boosting による結果は F 値で 94.56，ADTrees による結果は F 値で 93.10 であった．SVMs を利用した場合はほぼ完璧に CRL 固有表現データを分類しており，2 節で述べた手法 (b) の欠点を示していると言える．この学習データからは SVMs のラベルをそのまま出力するという分類器が作成され，Stacking がうまく働かなかったと予想される．

手法 (c) の評価結果は表 7〜表 9 である．手法 (c) では，CRL 固有表現データを記事単位で N 分割し，N-1 をそれぞれの分類器作成用，残り 1 を Stacking 用として学習を行なった．手法 (c) では，学習データを N 分割した場合，N 個の結果が得られるため，それぞれの条件で最も良かった結果だけを載せている．表 7〜表 9 で，全体の F 値で表 2 より良い結果を残せたのは，表 7 にある SVM-Boosting-ADTrees の組み合わせだけであった．手法 (c) による実験では，全部で，140 ((5 + 10 + 20) × 4) の実験結果が得られており，全体の F 値で表 2 より良い結果となったのは，140 実験結果中 2 つだけであった．この実験結果から，2 節で述べたように，手法 (c) では分類器作成用の学習データと Stacking 用の学習データの良い比率見つけることが難しいことわかる．

提案手法の評価結果は表 10〜表 12 である．提案手法の評価は，CRL 固有表現データを記事単位で 5 分割，10 分割，20 分割の 3 種類の条件で Stacking 用の学習

[5] ともに，http://www.csl.sony.co.jp/person/sekine/IREX/Package/IREXfinalB.tar.gz から入手可能．

表 5: 手法 (a) による NE 抽出の評価結果

NE	SVM-Boosting				SVM-ADTrees				Boosting-ADTrees				SVM-Boosting-ADTrees			
	Rec	Prec	F-M	vsSVM	Rec	Prec	F-M	vsSVM	Rec	Prec	F-M	vsSVM	Rec	Prec	F-M	vsSVM
ART	30.61	50	37.97	-1.59	28.57	46.67	35.44	-4.12	28.57	53.85	37.33	-2.23	26.53	43.33	32.91	-6.65
DATE	91.34	94.4	92.84	-1.69	89.89	92.57	91.21	-3.32	88.45	92.11	90.24	-4.29	92.78	96.98	94.83	**0.3**
LOC	80.77	84.85	82.76	-1.6	80.53	87.47	83.85	-0.51	80.29	84.13	82.16	-2.2	81.49	86.04	83.7	-0.66
MON	100	100	100	0	100	100	100	0	100	100	100	0	100	100	100	0
ORG	71.72	84.29	77.5	-1.58	71.72	84.8	77.72	-1.36	69.15	83.54	75.67	-3.41	72.24	84.13	77.73	-1.35
PERC	95.24	100	97.56	0	90.48	100	95	-2.56	90.48	95	92.68	-4.88	95.24	95.24	95.24	-2.32
EPRS	87.04	88.03	87.54	-1.04	87.89	88.64	88.26	-0.32	85.92	89.18	87.52	-1.06	87.61	89.37	88.48	-0.1
TIME	94.92	98.25	96.55	0	94.92	98.25	96.55	0	96.61	98.28	97.44	**0.89**	96.61	98.28	97.44	**0.89**
All	81.15	87.4	84.16	-1.18	80.9	87.96	84.28	-1.06	79.57	87	83.12	-2.22	81.78	88.26	84.9	-0.44

表 6: 手法 (b) による NE 抽出の評価結果

NE	SVM-Boosting				SVM-ADTrees				Boosting-ADTrees				SVM-Boosting-ADTrees			
	Rec	Prec	F-M	vsSVM	Rec	Prec	F-M	vsSVM	Rec	Prec	F-M	vsSVM	Rec	Prec	F-M	vsSVM
ART	36.73	42.86	39.56	0	36.73	42.86	39.56	0	28.57	34.15	31.11	-8.45	36.73	42.86	39.56	0
DATE	93.5	95.57	94.53	0	93.5	95.57	94.53	0	91.7	94.07	92.87	-1.66	93.5	95.57	94.53	0
LOC	81.01	87.99	84.36	0	81.01	87.99	84.36	0	80.53	85.9	83.13	-1.23	81.01	87.99	84.36	0
MON	100	100	100	0	100	100	100	0	100	100	100	0	100	100	100	0
ORG	74.81	83.86	79.08	0	74.81	83.86	79.08	0	72.49	82.94	77.37	-1.71	74.81	83.86	79.08	0
PERC	95.24	100	97.56	0	95.24	100	97.56	0	100	95.45	97.67	0.11	95.24	100	97.56	0
EPRS	88.45	88.7	88.58	0	88.45	88.7	88.58	0	86.76	88.51	87.62	-0.96	88.45	88.7	88.58	0
TIME	94.92	98.25	96.55	0	94.92	98.25	96.55	0	96.61	96.61	96.61	0.06	94.92	98.25	96.55	0
All	82.86	87.98	85.34	0	82.86	87.98	85.34	0	81.34	86.6	83.89	-1.45	82.86	87.98	85.34	0

データを獲得して行なった．表 10 が 5 分割，表 11 が 10 分割，表 12 が 20 分割での結果である．全部で 12 (=3 × 4) の実験結果が得られ，全体の F 値においては，12 実験結果中の 5 実験結果がベースラインより良く，1 実験結果はベースラインと同じであった．

比較対象である手法 (a)〜(c) の中でベースラインを越える結果を残せたのは手法 (c) だけであり，手法 (c) でベースライン以上となったのは 140 実験結果中わずか 2 実験結果だけであった．それに対し，本手法は 12 実験結果のうち 6 実験結果がベースライン以上であり，手法 (a)〜(c) と比べ効果的な学習方法であると言える．それぞれの項目別に抽出結果を見てみると，Stacking なしでも比較的高精度が得られている DATE，MONEY，PERCENT，TIME においてはわずかながら精度が下がる傾向にある．一方，ARTIFACT や LOCATION においては改善が目立つ．Stacking は難しい分類問題に対しては有効に働くものの簡単な分類問題にはうまく働かないと予想される．

図 4 は，それぞれの分類器の組み合わせでの各分割数での全体の F 値の変化を示している．手法 (a)，手法 (c) は最も良かった結果を載せている．手法 (b) に関しては，最も良い結果がベースラインと同じであったため省略している．本手法では，SVM-Boosting の組み合わせを除いては，分割数を細かくすることで徐々に精度が上昇していき，SVMs-ADTrees，SVMs-Boosting-ADTrees の組み合わせはベースラインを越えている．この結果から，本手法では，分割数を細かくして交差検定を行なって Stacking 用の学習データを獲得することで，精度を改善できることがわかる．SVM-Boosting の組み合わせがうまくいかない理由としては，それぞれの全体の F 値の差が 3 ポイントほどあるため，Boosting による分類結果が Stacking にうまく働かなかったためだと予想される．

SVM-ADTrees の組み合わせは，20 分割から得られた学習データから Stacking 学習を行なった結果，F 値で 85.83 まで上昇した．この結果は，磯崎ら [10] が報告している CRL データを学習データとした GENERAL に対する抽出結果の F 値で 85.77 より良い結果である．磯崎らは素性を拡張するために NTT 日本語語彙大系 [11] を利用しているが，本手法では NTT 日本語語彙大系のような言語資源を利用せずとも精度を上回ることができた．

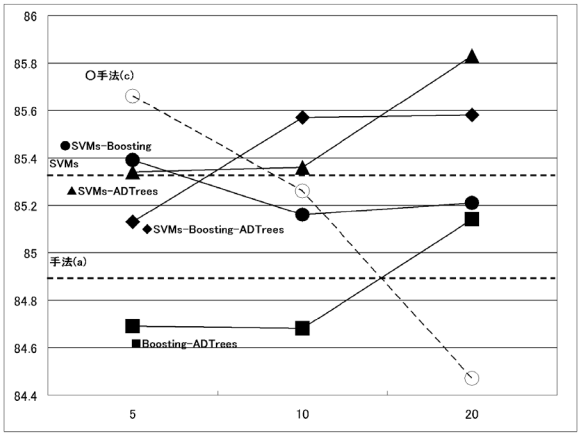

図 4: 分割数による全体の F 値の変化 (横軸:分割数，縦軸:全体の F 値．実線：本手法，破線：比較対象．)

5 関連研究

今回，交差検定を利用して，Stacking 学習学習データ作成し，Stacking 学習を行なう手法を提案した．この類似研究として，学習データの自動獲得による Post Editing 規則学習と Error Correction の研究がある．

中川ら [5] は，形態素解析器が出力するラティスに対し，正解・誤り二値分類器を Post Editing として適用し，最終的な解を得る手法を提案している．彼らの手法では，Post Editing 規則学習のための学習データ獲得のために形態素解析器作成のための学習データに対

表 7: 手法 (c) による NE 抽出の評価結果．4/5 を 1 段目の分類器作成，1/5 を Stacking 学習に利用

NE	SVM-Boosting				SVM-ADTrees				Boosting-ADTrees				SVM-Boosting-ADTrees			
	Rec	Prec	F-M	vsSVM	Rec	Prec	F-M	vsSVM	Rec	Prec	F-M	vsSVM	Rec	Prec	F-M	vsSVM
ART	36.73	50	42.35	**2.79**	34.69	48.57	40.48	**0.92**	24.49	32.43	27.91	-11.65	32.65	48.48	39.02	-0.54
DATE	90.97	96.18	93.51	-1.02	92.06	95.86	93.92	-0.61	89.53	96.5	92.88	-1.65	91.7	96.58	94.07	-0.46
LOC	81.49	86.92	84.12	-0.24	81.97	86.99	84.41	**0.05**	83.17	86.93	85.01	**0.65**	82.93	88.46	85.61	**1.25**
MON	100	100	100	0	100	100	100	0	100	100	100	0	100	100	100	0
ORG	74.04	84.46	78.9	-0.18	73.26	84.82	78.62	-0.46	72.75	83.73	77.85	-1.23	73.78	84.16	78.63	-0.45
PERC	80.95	100	89.47	-8.09	85.71	100	92.31	-5.25	85.71	100	92.31	-5.25	85.71	100	92.31	-5.25
EPRS	88.17	88.92	88.54	-0.04	88.73	88.73	88.73	0.15	87.04	90.35	88.67	**0.09**	88.45	89.97	89.2	**0.62**
TIME	94.92	98.25	96.55	0	94.92	98.25	96.55	0	96.61	96.61	96.61	**0.06**	94.92	98.25	96.55	0
All	82.1	88.3	85.09	-0.25	82.35	88.33	85.24	-0.1	81.47	87.98	84.6	-0.74	82.54	89.02	85.66	**0.32**

表 8: 手法 (c) による NE 抽出の評価結果．9/10 を 1 段目の分類器作成，1/10 を Stacking 学習に利用

NE	SVM-Boosting				SVM-ADTrees				Boosting-ADTrees				SVM-Boosting-ADTrees			
	Rec	Prec	F-M	vsSVM	Rec	Prec	F-M	vsSVM	Rec	Prec	F-M	vsSVM	Rec	Prec	F-M	vsSVM
ART	34.69	43.59	38.64	-0.92	28.57	58.33	38.36	-1.2	18.37	34.62	24	-15.56	36.73	52.94	43.37	**3.81**
DATE	93.86	95.94	94.89	**0.36**	92.06	95.86	93.92	-0.61	91.7	96.21	93.9	-0.63	92.42	95.88	94.12	-0.41
LOC	82.45	88.63	85.43	**1.07**	81.25	88.25	84.61	**0.25**	81.01	84.04	82.5	-1.86	81.73	88.08	84.79	**0.43**
MON	86.67	76.47	81.25	-18.75	100	100	100	0	86.67	76.47	81.25	-18.75	100	100	100	0
ORG	73.01	81.61	77.07	-2.01	72.49	82.46	77.15	-1.93	72.24	84.89	78.06	-1.02	73.52	82.66	77.82	-1.26
PERC	90.48	100	95	-2.56	90.48	100	95	-2.56	100	95.45	97.67	**0.11**	90.48	100	95	-2.56
EPRS	88.45	89.2	88.83	**0.25**	88.17	89.17	88.67	**0.09**	88.17	90.72	89.43	**0.85**	87.61	89.63	88.6	**0.02**
TIME	94.92	100	97.39	**0.84**	94.92	98.25	96.55	0	91.53	93.1	92.31	-4.24	94.92	98.25	96.55	0
All	82.61	87.71	85.08	-0.26	81.72	88.68	85.06	-0.28	81.09	87.57	84.2	-1.14	82.29	88.44	85.26	-0.08

し作成した形態素解析器で解析を行ない，ラティス内の確率値でソートされた形態素解析結果の候補のうち最初の正解が出現するまでの誤りを誤り事例，正しい結果は正解事例としてすることで，Post Editing 用の学習データを作成している．この手法は，SVMs と比べ性能の低い HMMs を利用しているために，上手く学習データを獲得を獲得できていると予想されるが，SVMs のような機械学習アルゴリズムでは上手くいかないと予想される．

本論文では，Stacking 学習のための学習データを交差検定を使って獲得しているのに対し，Wu[9] らは Error Correction のための学習データを交差検定を使って獲得している．この手法は，本手法で採用した学習データ獲得手法と同様に，SVMs のように，学習データに対するテストでは，かなりの高精度で分類できるモデルを生成できる学習器を利用する場合にも有効であると考えられる．

6 まとめ

交差検定の枠組を利用し，Stacking を効率的に学習できる手法を提案し，日本語固有表現抽出での評価実験を行なった．実験結果から，CRL 固有表現データを学習データとし，GENERAL で評価したところ，単独の機械学習アルゴリズムおよび本実験で比較対象とした 3 種類の Stacking 学習手法による実験結果よりも良い結果を残すことができた．

本提案手法は細かい分割単位で Stacking 学習データを獲得することで抽出精度を改善でき，一番良い結果は F 値で 85.83 となった．

本提案手法は，日本語固有表現だけでなく，様々なタスクにおいても，精度向上が期待される．今後の課題としては，日本語固有表現抽出以外での評価，Voting など他の手法との比較が必要である．

謝辞

今回の実験にあたり，奈良先端大学院大学で開発された，ChaSen，YamCha を利用させていただきました．深く感謝いたします．

参考文献

[1] IREX ワークショプ予稿集. IREX 実行委員会, 1999.

[2] Masayuki Asahara and Yuji Matsumoto. Japanese named entity extraction with redundant morphological analysis. In *Proceedings of Human Language Technology and North American Chapter of Association for Computational Linguistics*, pp. 8–15, 2003.

[3] Radu Florian, Abe Ittycheriah, Hongyan Jing, and Tong Zhang. Named entity recognition through classifier combination. In *Proceedings of CoNLL-2003*, pp. 168–171, 2003.

[4] Yoav Freund and Llew Mason. The alternating decision tree learning algorithm, 1999.

[5] Tetsuji Nakagawa, Taku Kudo, and Yuji Matsumoto. Revision learning and its application to part-of-speech tagging. In *Proceedings of 40th Annual Meeting of Association for Computational Linguistics*, pp. 497–504, 2002.

[6] Robert E. Schapire and Yoram Singer. Boostexter: A boosting-based system for text categorization. *Machine Learning*, Vol. 39(2/3), pp. 135–168, 2000.

[7] Takehito Utsuro, Manabu Sassano, and Kiyotaka Uchimoto. Combining outputs of multiple Japanese named entity chunkers by stacking. In *Proceedings of EMNLP 2002*, pp. 281–288, 2002.

[8] Vladimir Vapnik. *Statistical Learning Theory*. John Wiley & Sons, 1998.

[9] Dekai Wu, Grace Ngai, and Marine Carpuat. N-fold templated piped correction. In *First International Joint Conference on Natural Language Processing*, pp. 632–637, 2004.

[10] 磯崎秀樹, 賀沢秀人. SVM に基づく固有表現抽出の高速化. In *IPSJ SIG notes NL-149-1*, pp. 1–8, 2002.

[11] 池原悟, 宮崎正弘, 白井諭, 横尾昭男, 中岩浩巳, 小倉健太郎, 大山芳史, 林良彦. 日本語語彙大系：CD-ROM 版. 岩波書店, 1997.

表 9: 手法 (c) による NE 抽出の評価結果．19/20 を 1 段目の分類器作成，1/20 を Stacking 学習に利用

NE	SVM-Boosting				SVM-ADTrees				Boosting-ADTrees				SVM-Boosting-ADTrees			
	Rec	Prec	F-M	vsSVM	Rec	Prec	F-M	vsSVM	Rec	Prec	F-M	vsSVM	Rec	Prec	F-M	vsSVM
ART	26.53	46.43	33.77	-5.79	24.49	54.55	33.8	-5.76	2.04	11.11	3.45	-36.11	16.33	33.33	21.92	-17.64
DATE	92.78	97.35	95.01	**0.48**	92.06	95.51	93.75	-0.78	91.34	93.7	92.5	-2.03	92.78	97.72	95.19	**0.66**
LOC	81.49	88.28	84.75	**0.39**	81.01	87.53	84.14	-0.22	80.05	86.05	82.94	-1.42	82.45	88.17	85.22	**0.86**
MON	86.67	76.47	81.25	-18.75	86.67	76.47	81.25	-18.75	100	100	100	0	86.67	76.47	81.25	-18.75
ORG	73.78	81.3	77.36	-1.72	74.81	82.44	78.44	-0.64	71.21	86.02	77.92	-1.16	73.26	82.13	77.45	-1.63
PERC	80.95	100	89.47	-8.09	80.95	100	89.47	-8.09	100	95.45	97.67	**0.11**	80.95	100	89.47	-8.09
EPRS	88.17	90.2	89.17	**0.59**	88.17	88.42	88.29	-0.29	86.2	88.7	87.43	-1.15	87.61	90.14	88.86	**0.28**
TIME	94.92	98.25	96.55	0	93.22	100	96.49	-0.06	96.61	100	98.28	**1.73**	94.92	98.25	96.55	0
All	81.91	88.28	84.97	-0.37	81.78	87.96	84.76	-0.58	79.89	88.51	83.98	-1.36	81.59	88.42	84.87	-0.47

表 10: 提案手法の評価結果．5 分割の交差検定から獲得した Stacking 学習データを利用

NE	SVM-Boosting				SVM-ADTrees				Boosting-ADTrees				SVM-Boosting-ADTrees			
	Rec	Prec	F-M	vsSVM	Rec	Prec	F-M	vsSVM	Rec	Prec	F-M	vsSVM	Rec	Prec	F-M	vsSVM
ART	42.86	44.68	43.75	**4.19**	40.82	46.51	43.48	**3.92**	38.78	40.43	39.58	**0.02**	40.82	44.44	42.55	**2.99**
DATE	92.78	94.83	93.8	-0.73	90.61	92.96	91.77	-2.76	90.61	94.36	92.45	-2.08	90.97	92.99	91.97	-2.56
LOC	81.97	88.34	85.04	**0.68**	81.97	89.97	85.79	**1.43**	81.97	88.57	85.14	**0.78**	81.97	89.03	85.36	**1**
MON	100	100	100	0	100	100	100	0	100	100	100	0	100	100	100	0
ORG	74.55	82.39	78.27	-0.81	75.32	83	78.98	-0.1	74.04	82.05	77.84	-1.24	74.81	81.74	78.12	-0.96
PERC	100	95.45	97.67	**0.11**	95.24	95.24	95.24	-2.32	100	95.45	97.67	**0.11**	100	95.45	97.67	**0.11**
PERS	88.45	89.97	89.2	**0.62**	88.73	89.49	89.11	**0.53**	87.04	89.83	88.41	-0.17	88.73	90.26	89.49	**0.91**
TIME	94.92	98.25	96.55	0	93.22	98.21	95.65	-0.9	94.92	94.92	94.92	-1.63	94.92	96.55	95.73	-0.82
All	83.18	87.73	85.39	**0.05**	82.86	87.98	85.34	0	82.23	87.31	84.69	-0.65	82.92	87.46	85.13	-0.21

表 11: 提案手法の評価結果．10 分割の交差検定から獲得した Stacking 学習データを利用

NE	SVM-Boosting				SVM-ADTrees				Boosting-ADTrees				SVM-Boosting-ADTrees			
	Rec	Prec	F-M	vsSVM	Rec	Prec	F-M	vsSVM	Rec	Prec	F-M	vsSVM	Rec	Prec	F-M	vsSVM
ART	40.82	40	40.4	**0.84**	40.82	43.48	42.11	**2.55**	40.82	36.36	38.46	-1.1	42.86	42.86	42.86	**3.3**
DATE	92.78	95.54	94.14	-0.39	92.06	94.44	93.24	-1.29	91.7	96.58	94.07	-0.46	92.78	95.19	93.97	-0.56
LOC	81.49	88.28	84.75	**0.39**	81.25	89.66	85.25	**0.89**	81.73	88.54	85	**0.64**	82.69	89.82	86.11	**1.75**
MON	100	100	100	0	100	100	100	0	100	100	100	0	100	100	100	0
ORG	74.29	82.57	78.21	-0.87	75.32	83.48	79.19	**0.11**	73.78	82.23	77.78	-1.3	74.55	81.69	77.96	-1.12
PERC	100	95.45	97.67	**0.11**	95.24	95.24	95.24	-2.32	100	95.45	97.67	**0.11**	100	95.45	97.67	**0.11**
PERS	88.45	89.46	88.95	**0.37**	88.73	88.73	88.73	**0.15**	87.04	89.05	88.03	-0.55	88.73	90.26	89.49	**0.91**
TIME	94.92	98.25	96.55	0	94.92	96.55	95.73	-0.82	94.92	94.92	94.92	-1.63	94.92	94.92	94.92	-1.63
All	82.92	87.52	85.16	-0.18	82.99	87.88	85.36	**0.02**	82.35	87.15	84.68	-0.66	83.43	87.82	85.57	**0.23**

表 12: 提案手法の評価結果．20 分割の交差検定から獲得した Stacking 学習データを利用

NE	SVM-Boosting				SVM-ADTrees				Boosting-ADTrees				SVM-Boosting-ADTrees			
	Rec	Prec	F-M	vsSVM	Rec	Prec	F-M	vsSVM	Rec	Prec	F-M	vsSVM	Rec	Prec	F-M	vsSVM
ART	40.82	40	40.4	**0.84**	40.82	40.82	40.82	**1.26**	38.78	39.58	39.18	-0.38	36.73	38.3	37.5	-2.06
DATE	91.7	95.85	93.73	-0.8	93.5	95.93	94.7	**0.17**	92.06	96.59	94.27	-0.26	92.78	96.25	94.49	-0.04
LOC	81.49	88.51	84.86	**0.5**	83.65	90.63	87	**2.64**	82.45	87.95	85.11	**0.75**	83.41	88.97	86.1	**1.74**
MON	100	100	100	0	100	100	100	0	100	100	100	0	100	100	100	0
ORG	75.06	82.49	78.6	-0.48	75.06	83.43	79.03	-0.05	74.55	83.09	78.59	-0.49	75.32	83	78.98	-0.1
PERC	100	95.45	97.67	**0.11**	95.24	95.24	95.24	-2.32	100	95.45	97.67	**0.11**	95.24	95.24	95.24	-2.32
PERS	88.73	89.74	89.24	**0.66**	87.32	89.08	88.19	-0.39	87.32	89.08	88.19	-0.39	87.61	89.63	88.6	**0.02**
TIME	93.22	98.21	95.65	-0.9	94.92	96.55	95.73	-0.82	94.92	96.55	95.73	-0.82	94.92	98.25	96.55	0
All	82.92	87.63	85.21	-0.13	83.49	88.29	85.83	**0.49**	82.8	87.62	85.14	-0.2	83.3	87.98	85.58	**0.24**