ニューラルネットワークを用いた系列ラベリングによる単語分割手法

篠原　正太†　　山名　早人††,†††

† 早稲田大学基幹理工学研究科　〒169–8555 東京都新宿区大久保 3-4-1
†† 早稲田大学基幹理工学術院　〒169–8555 東京都新宿区大久保 3-4-1
††† 国立情報学研究所　〒101–8430 東京都千代田区一ツ橋 2-1-2
E-mail: †{shinohara,yamana}@yama.info.waseda.ac.jp

あらまし　日本語を対象とした自然言語処理では多くの場合，事前に形態素解析により意味を持つ最小の単位である形態素に分割され特徴量として利用される．形態素解析処理には単語ラティス上の経路予測や点予測に基づく手法が広く用いられている．また形態素解析を単語分割と品詞推定の 2 つの処理に分けて，それぞれについても研究が行われている．昨今の単語分割に関する研究では，ニューラルネットワークを用いた系列ラベリングによる手法が多く提案されている．しかし，ニューラルネットワークのモデルの過学習や，小規模なコーパスにおける学習では有用な特徴量が得られない問題がある．そこで本研究では，ニューラルネットワークを用いた系列ラベリングによる手法をベースとして，コーパスの規模に依存しない単語分割手法を提案する．5 つのデータセットを用いた評価実験の結果，既存のニューラルネットワークを用いた手法に対して提案手法では，F_1 が平均 94.43 から 96.00 へ向上したことにより有効性を確認した．

キーワード　自然言語処理，テキストマイニング

1. はじめに

日本語を対象とした自然言語処理では多くの場合，事前にテキストデータを形態素解析により形態素ごとに分割し特徴量として利用する．形態素とは，言語で意味を持つ最小単位のことであり，形態素解析ではテキストデータを形態素ごとに分割し，それぞれの形態素の読みや原形，品詞，活用の種類を判別する．自然言語のテキストデータを形態素ごとに分割することで，文や文書を形態素の集合や系列データとして扱うことができる．それにより，文や文書を高次元のベクトルで表現でき，文書分類や機械翻訳など様々なタスクに適用することができる．また系列データとして扱うことで，言語モデルなどに応用することができる．

一般的に形態素解析処理には，1) 単語ラティス上の経路予測 [1] [3] [4] や，2) 点予測 [5] に基づく手法が広く用いられている．単語ラティス上の経路予測とは，単語の候補を列挙したグラフ構造である単語ラティスに対し，最適な経路を予測することで形態素解析を行う手法である．また点予測とは，単語の境界を分類するための素性として，周囲の単語境界や品詞等の推定値を利用せずに，周囲の文字列の情報のみを利用する手法である．実際に，既製の形態素解析ツールである MeCab (注1) や JUMAN (注2) には単語ラティス上の経路予測，KyTea (注3) には点予測に基づく手法が用いられている．MeCab や JUMAN では，単語ラティスの生成の際に辞書と呼ばれる単語の品詞等の情報が必要となる．これらの解析精度を高めるためには，辞書のメンテナンスやコーパスのアノテーションが必要となり，人手でこれらを行う場合コストがかかる．

また，形態素解析を単語分割と品詞推定の 2 つの処理に細分化して，それぞれについても研究が行われている．昨今の単語分割に関する研究では，ニューラルネットワークを適用した研究が多くなされている．ニューラルネットワークのモデルでは，ネットワーク構造によるハイパーパラメータのチューニングが必要となるが，高次元で疎な特徴量から低次元で密な特徴量を獲得することができる．これにより，従来の特徴量エンジニアリングを自動化し，有用な特徴量の獲得が期待できる．ニューラルネットワークを適用した研究の中でも，特に系列ラベリング問題として扱う手法が多く提案されている [2] [6] [7] ．系列ラベリングによる単語分割では，入力文の各文字の系列に対して，S(Simple), B(Beginning), M(Middle), E(End) などのラベル列を推定することによって分割を行う．しかし，既存研究ではニューラルネットワークのモデルの過学習や，小規模なコーパスにおける学習では有用な特徴量が得られない問題がある．

そこで本研究では，ニューラルネットワークを利用して系列ラベリング問題として扱う手法をベースとして，過学習を回避し，コーパスの規模に依存しない単語分割手法を提案する．提案手法では，モデルの学習時に，入力文の一部の文字を学習時に出現していない未知の文字として学習することで，過学習を回避し，小規模なコーパスに対しても有用な特徴量の獲得を図る．

本稿は以下の構成をとる．まず，2. で関連研究について述べ，

(注1)：MeCab, http://taku910.github.io/mecab/ （2016/12/09 アクセス）
(注2)：日本語形態素解析システム JUMAN, http://nlp.ist.i.kyoto-u.ac.jp/index.php?cmd=read&page=JUMAN （2016/12/09 アクセス）
(注3)：京都テキスト解析ツールキット KyTea, http://www.phontron.com/kytea/index-ja.html （2016/12/09 アクセス）

3. で提案手法について説明する．そして，4. で，実験・評価を行い，最後に，5. でまとめる．

2. 関連研究

本節では，本研究に関連する研究について説明する．2.1 では形態素解析における高速な単語ラティス生成手法，2.2 では深層ニューラルネットワークによる日本語単語分割手法について説明する．

2.1 形態素解析における高速な単語ラティス生成手法

鍛冶ら [1] は，解析精度を犠牲にすることなく高速に単語ラティスを生成する手法を提案した．従来の単語ラティスの生成方法には，辞書引きと文字種に基づくヒューリスティックの組み合わせや，単語ラティス生成のための枝刈りアルゴリズムなどがあるが，前者は未知語に対して脆弱であり，後者は $\mathcal{O}(n^2)$ の時間計算量が発生してしまう問題がある．

そこで，鍛冶らが提案した段階的単語ラティス生成アルゴリズムでは，単語ラティスを構成する単語と品詞タグを独立に生成する．単語分割モデルでは，系列ラベリングにより単語分割を行っており，学習には構造化パーセプトロンを用いている．品詞タグ生成モデルは，多クラス線形モデルを構築して，学習には平均化パーセプトロンを用いている．単語分割モデルと品詞タグ生成モデルはそれぞれ $\mathcal{O}(n)$ で計算できる．そのため，単語分割モデルを用いて，スコア上位 α 件の単語分割結果の和集合 W を求め，品詞タグ生成モデルを用いて，各単語 $w \in W$ に対してスコア上位 β 件の品詞タグ集合 T を生成した場合でも，$\mathcal{O}(\alpha\beta n)$ の時間計算量で計算できる．また生成された単語ラティスについても，正解の単語分割結果のカバー率を下げることなく，枝刈りアルゴリズムと比べて単語ラティスのサイズを減らすことができた．

ここで鍛冶らは，全体の解析精度を上げるために単語分割モデルの特徴量として辞書を用いているため，辞書に登録されていない単語に対して脆弱になると考えられる．

2.2 深層ニューラルネットワークによる日本語単語分割

北川ら [2] は，深層ニューラルネットワークを利用した日本語単語分割手法を提案した．これまで，深層ニューラルネットワークを用いた中国語の単語分割の手法は数多く提案されていた [6] [7] ．そこで，日本語単語分割における深層ニューラルネットワークの構造による特性を調査し性能の評価を行った．

北川らは，単語分割を系列ラベリング問題として扱い，周辺の文字列を入力として，対象の文字のラベルの確率分布を出力とするニューラルネットワークモデルを用いた．また日本語には，中国語などとは異なり，平仮名やカタカナなどの文字種が存在するため，文字 embedding と文字種 embedding を結合したベクトルを入力とした．評価実験の結果，既成のソフトウェアである KyTea ほどの精度は出ないまでも，学習データが十分にある場合には，同程度の精度で解析ができることが分かった．

北川らの研究から，日本語に深層ニューラルネットワークを適用した場合，従来の特徴量エンジニアリングによる手法に比べて有用な特徴量が得られにくい点や，小規模なコーパスにおいて学習が困難であり性能が劣化する点などが問題点として挙げられる．

3. 提案手法

本節では，ニューラルネットワークにおいて過学習の問題や小規模なコーパスにおいて学習が困難であることに対して，コーパスの規模に依存せず学習可能な単語分割手法について述べる．まず，3.1 で提案手法の概要について説明する．次に，3.2 で提案手法であるニューラルネットワークを用いた系列ラベリングによる単語分割について説明する．

3.1 概　　要

2. において，関連研究として単語分割および形態素解析の研究について述べた．しかし，それぞれの研究において以下の問題点が存在する．

1) 特徴量に辞書を用いる場合，辞書にない単語に対して脆弱 [1]
2) 過学習の問題や小規模なコーパスにおける学習が困難 [2]

提案手法では，これらの問題について以下のように対応する．

1) 辞書を用いずに周辺の文字と文字種のみを入力とすることで，辞書にない単語に対しても頑健に分割できるようにする．
2) 入力文字をある確率に従って未知の文字として学習することで，未知の入力に対しても頑健なモデルの学習や過学習の回避を図る．

具体的な提案手法は以下の手順に沿う．

1. 系列ラベリング問題として扱うため，周辺の文字と文字種を入力として，ニューラルネットワークモデルにより，対象の文字に対する各ラベル S, B, M, E の生起確率を求める．
2. 得られたラベルの生起確率と学習データより求めたラベルの遷移確率を用いて，動的計画法の Viterbi アルゴリズムにより，入力文字列に対する最適なラベル列を求める．
3. ラベル列より単語分割結果を得る．

3.2 ニューラルネットワークを用いた系列ラベリングによる単語分割

本項では，提案手法であるニューラルネットワークを用いた系列ラベリングによる単語分割について説明する．まず，3.2.1 で系列ラベリングによる単語分割について説明する．次に，3.2.2 で動的計画法による最適なラベルの選択方法について説明する．最後に，3.2.3 で確率的文字種スケーリングによる効果的な学習方法について説明する．

3.2.1 系列ラベリングによる単語分割

提案手法では，単語分割を系列ラベリング問題として扱う．それぞれの文字には，S(Simple), B(Beginning), M(Middle), E(End) のいずれかのラベルが付けられる．ここで，S は 1 文字で 1 単語であること，B, M, E はそれぞれ，2 文字以上で 1 単語である文字列の先頭，中間，終端を表す．

提案手法で用いたニューラルネットワークの構造を図 1 に示す．ニューラルネットワークモデルでは，各文字を入力として 4 種類のラベルの生起確率を出力する．文字数 n の入力文 $c_{1:n}$ の各文字 c_i $(1 \leqq i \leqq n)$ に対する入力は，ウィンドウサイズを k $(k \in \mathbb{N}^{odd} = \{\ x \mid x \text{ は奇数}\ \})$ とすると，$(c_{i-\frac{k-1}{2}}, \ldots, c_i, \ldots, c_{i+\frac{k-1}{2}})$ となる．また文頭と文末にはそれぞれ文頭記号と文末記号を付加する．

まず，入力である $(c_{i-\frac{k-1}{2}}, \ldots, c_i, \ldots, c_{i+\frac{k-1}{2}})$ のそれぞれの文字を，実数値ベクトルである embedding に変換する．文字 c_i に対する次元数 d の embedding $\boldsymbol{x_i} \in \mathbb{R}^d$ は，次元数 d_c の文字 embedding $\boldsymbol{x}_{c_i} \in \mathbb{R}^{d_c}$ と次元数 d_p の文字種 embedding $\boldsymbol{x}_{p_i} \in \mathbb{R}^{d_p}$ を連結したベクトルであり，式 (1) により表現する．

$$\boldsymbol{x_i} = \begin{bmatrix} \boldsymbol{x}_{c_i} \\ \boldsymbol{x}_{p_i} \end{bmatrix} \tag{1}$$

ここで，$d = d_c + d_p$ である．文字集合を C，その次元を $|C|$ とすると，文字 $c \in C$ に対する文字 embedding $\boldsymbol{x}_c \in \mathbb{R}^{d_c}$ は，ルックアップテーブル $\boldsymbol{M}_c \in \mathbb{R}^{d_c \times |C|}$ によりマッピングされる．同様に，文字種集合を P，その次元を $|P|$ とすると，文字種 $t \in T$ に対する文字種 embedding $\boldsymbol{x}_p \in \mathbb{R}^{d_p}$ は，ルックアップテーブル $\boldsymbol{M}_p \in \mathbb{R}^{d_p \times |P|}$ によりマッピングされる．提案手法では文字種として，平仮名，カタカナ，漢字，数字，アルファベット，その他，文頭，文末の 8 種類を用いた．これにより，入力 $(c_{i-\frac{k-1}{2}}, \ldots, c_i, \ldots, c_{i+\frac{k-1}{2}})$ に対する embedding $(\boldsymbol{x}_{i-\frac{k-1}{2}}, \ldots, \boldsymbol{x}_i, \ldots, \boldsymbol{x}_{i+\frac{k-1}{2}})$ が得られる．

次に，得られた embedding を隠れ層の 1 層のノードとして，隣接する 2 つのノードを連結することにより 2 層のノードを求めていく．隣接する 2 つのノードを $\boldsymbol{h}_L, \boldsymbol{h}_R \in \mathbb{R}^d$ とすると連結したノード $\boldsymbol{h}_P \in \mathbb{R}^d$ は式 (2) により表現できる．

$$\boldsymbol{h}_P = g(\boldsymbol{W}_1 \begin{bmatrix} \boldsymbol{h}_L \\ \boldsymbol{h}_R \end{bmatrix} + \boldsymbol{b}_1) \tag{2}$$

ここで，$\boldsymbol{W}_1 \in \mathbb{R}^{d \times 2d}$ は重み行列，$\boldsymbol{b}_1 \in \mathbb{R}^d$ はバイアス，$g(\cdot)$ は *sigmoid* などの非線形関数を表す．この操作を隠れ層のノードが 1 つになる，k 層まで繰り返し行う．したがって，隠れ層の数を $l \in [1, k]$ とすると，l 層の隠れ層の j 番目のノード $\boldsymbol{h}_j^l$ は式 (3) により表現できる．

$$\boldsymbol{h}_j^l = \begin{cases} g(\boldsymbol{W}_1 \begin{bmatrix} \boldsymbol{h}_j^{l-1} \\ \boldsymbol{h}_{j+1}^{l-1} \end{bmatrix} + \boldsymbol{b}_1) & (l \neq 1) \\ \boldsymbol{x}_{i-\frac{k+2j-3}{2}} & (l = 1) \end{cases} \tag{3}$$

そして，k 層の隠れ層のノード $\boldsymbol{h}_1^k$ を用いて，式 (4) により，

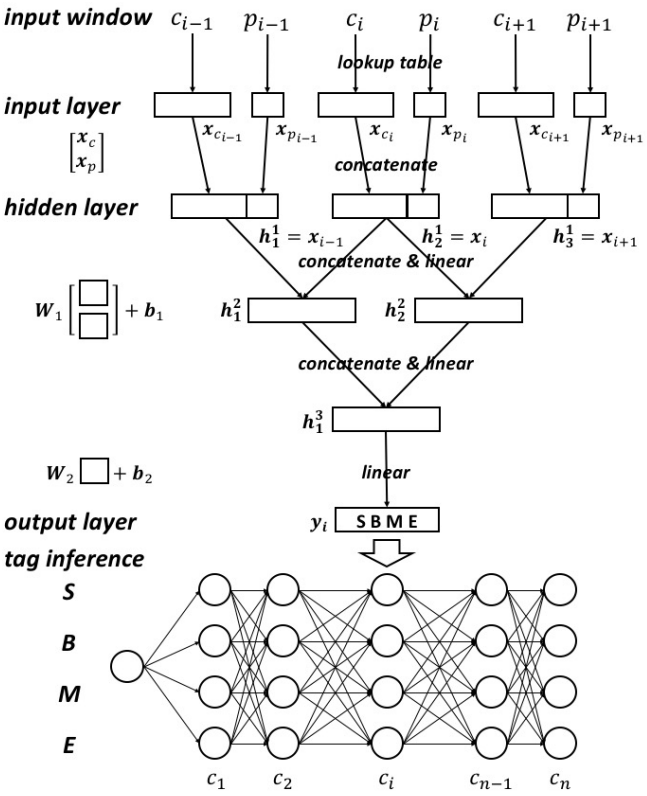

図 1　ニューラルネットワークを用いた単語分割

文字 c_i の各ラベルの生起確率 $\boldsymbol{y}_i \in \mathbb{R}^{|T|}$ を求める．

$$\boldsymbol{y}_i = softmax(\boldsymbol{W}_2 \boldsymbol{h}_1^k + \boldsymbol{b}_2) \tag{4}$$

ここで，$\{S, B, M, E\}$ のラベル集合を T，その次元を $|T|$ とすると，$\boldsymbol{W}_2 \in \mathbb{R}^{|T| \times d}$ は重み行列，$\boldsymbol{b}_2 \in \mathbb{R}^{|T|}$ はバイアスを表す．

ニューラルネットワークの学習の際には，出力 $\boldsymbol{y}_i$ と正解ラベル分布 $\boldsymbol{l}_i \in \mathbb{R}^{|T|}$ の交差エントロピー誤差を目的関数とする．交差エントロピー誤差関数を式 (5) に示す．

$$\boldsymbol{loss} = \sum_i -\boldsymbol{l}_i \log \boldsymbol{y}_i \tag{5}$$

この目的関数を最小化するように，誤差逆伝搬によりニューラルネットワークのパラメータである $\boldsymbol{M}_c, \boldsymbol{M}_p, \boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{b}_1, \boldsymbol{b}_2$ を更新する．

3.2.2 動的計画法による最適なラベルの選択

式 (4) によって求めたラベルの生起確率 $\boldsymbol{y}_i$ を用いて，入力文に対して最適化されたラベル列を求める方法について説明する．ラベル列の最適化には動的計画法の 1 つである Viterbi アルゴリズムを用いる．

i 番目までの文字列に対するラベル列を $s_{1:i} = s_1, \ldots, s_i$，$i$ 番目の文字のラベル s_i を k とした場合，ラベル列 $s_{1:i}$ を生成する最大確率を v_i^k と表す．このとき，$i+1$ 番目の文字のラベル s_{i+1} を l とすると，ラベル列 $s_{1:i+1}$ を生成する最大確率 v_{i+1}^l は漸化式を用いて，式 (6) により表すことができる．

$$v_{i+1}^l = \boldsymbol{y}_{i+1}(l) \max_{s_{1:i}} \{A_{kl} v_i^k\} \tag{6}$$

ここで，A_{kl} はラベル列が k から l へ遷移する確率を表す．これにより，動的計画法を用いてアルゴリズム 1 により入力文 $c_{1:n}$ に対する最適なラベル列 $s_{1:n}$ を求めることができる．

Algorithm 1 Viterbi アルゴリズム

Require: $c_{1:n}$
$\boldsymbol{y}_1 \Leftarrow LabelEstimator(c_1)$
for $j = 1$ **to** $|\boldsymbol{y}_1|$ **do**
  $v_1^j \Leftarrow \boldsymbol{y}_1(j)$ // ラベル列をメモ化
end for
for $i = 2$ **to** n **do**
  $\boldsymbol{y}_i \Leftarrow LabelEstimator(c_i)$
  for $j = 1$ **to** $|\boldsymbol{y}_i|$ **do**
    for each $v' \in v_{i-1}$ **do**
      if $v_i^j < v' \times y_i(j)$ **then**
        $v_i^j \Leftarrow v' \times y_i(j)$ // i 番目までのラベル列をメモ化
      end if
    end for
  end for
end for
return $\max(v_n)$ // メモ化したラベル列を返す

3.2.3 確率的文字種スケーリングによる学習

小規模なコーパスにおいても効果的にニューラルネットワークモデルの学習を行う，確率的文字種スケーリング (Probabilistic Character-type Scaling) について説明する．

式 (1) で説明したように，ニューラルネットワークへの入力は，各文字 c_i の周辺の文字列の embedding $(x_{i-\frac{k-1}{2}}, \ldots, x_i, \ldots, x_{i+\frac{k-1}{2}})$ である．しかし，確率的文字種スケーリングを用いた場合には，式 (7) のように各文字 c_i を確率 θ $(0 \leqq \theta \leqq 1)$ により，未知の文字 c_U として embedding $\boldsymbol{x}_i$ に変換する．

$$\boldsymbol{x}_i = \begin{cases} \begin{bmatrix} \boldsymbol{x}_{c_i} \\ \boldsymbol{x}_{p_i} \end{bmatrix} & (r_i > \theta) \\ \begin{bmatrix} \boldsymbol{x}_U \\ \boldsymbol{x}_{p_i} \end{bmatrix} & (r_i \leqq \theta) \end{cases} \tag{7}$$

ここで，$\boldsymbol{x}_U$ は未知の文字 c_U の embedding，r_i $(0 \leqq r_i \leqq 1)$ は各文字 c_i ごとに生成した一様乱数である．$\boldsymbol{x}_U$ は通常の文字と同様に，ルックアップテーブル $\boldsymbol{M}_c$ により設定する．これにより，文字と文字種情報の重み付けや動詞の活用語尾の変化などの学習などが期待でき，過学習を回避し小規模なコーパスにおける効果的な学習を図る．

4. 評価・実験

本節では，提案手法の実験と評価について述べる．まず，4.1 で評価実験に用いるデータセットについて説明する．次に，4.2 で各種パラメータの調整について説明する．そして，4.3 で実験結果について述べる．

評価指標として，単語分割結果の単語単位での適合率と再現率の調和平均である F_1 を用いた．また，実装には Chainer 1.17.0 [11] を用い，実験環境を表 1 に示す．

表 1 実験環境

項目	値
CPU	Intel Core i7-5820K
メインメモリサイズ	16GB
GPU	NVIDIA Quadro K5200
GPU メモリサイズ	8GB

表 2 各データセットに含まれる文数

	訓練	開発	評価
KC	30,608	4,027	3,764
KNBC	3,453	384	348
BCCWJ	47,547	6,586	5,741
KC(tiny)	3,060	402	376
BCCWJ(tiny)	4,754	658	574

表 3 各データセットで採用したパラメータ

	k	d_c	d_p	θ
KC	5	100	10	0.10
KNBC	5	100	10	0.25
BCCWJ	5	100	10	0.05
KC(tiny)	5	100	10	0.25
BCCWJ(tiny)	5	100	10	0.25

4.1 データセット

本稿では，評価実験に用いるデータセットとして，京都大学テキストコーパス version 4.0 [8]，京都大学 NTT ブログコーパス version 1.0 [9]，現代日本語書き言葉均衡コーパス [10] の 3 つを用いた．以下では，それぞれ KC, KNBC, BCCWJ と表記し，各コーパスは訓練，開発，評価用に分割して利用する．また，KC と BCCWJ において，訓練，開発，評価データの文数がそれぞれ 10%となるようにランダムサンプリングした小規模コーパス KC(tiny), BCCWJ(tiny) を用いた．各コーパスに含まれる文数を表 2 に示す．

4.2 各種パラメータの調整

ニューラルネットワークモデルのハイパーパラメータである，ウィンドウサイズ k，文字 embedding の次元数 d_c，文字種 embedding の次元数 d_p，確率的文字種スケーリングの閾値 θ の調整を行う．各種パラメータは，開発データセットにおいて F_1 が一番高かった値を採用した．ウィンドウサイズ k は，隠れ層が深くなり勾配消失により学習が進まなくなることを防ぐため，Chen ら [6] の設定である 5 を採用した．文字 embedding の次元数 d_c は 10, 20, 30, 50, 100 から，文字種 embedding の次元数 d_p は 5, 10, 15, 20 から，確率的文字種スケーリングの閾値 θ は 0.00, 0.05, 0.10, 0.15, 0.20, 0.25, 0.30 からそれぞれ選択した．d_c, d_p については，既存研究 [2] [6] [7] から上記の範囲で必要十分であると考えた．θ については 0.30 より高い値に設定すると，文字情報の欠落により十分学習できないため上記の範囲を用いた．それぞれのデータセットにおいて調整したパラメータを表 3 に示す．また最適化には Adam を利用しているが，パラメータは Chainer のデフォルト設定を用いた．

表 4　既存手法との F_1 の比較

	Method	KC	KNBC	BCCWJ	KC(tiny)	BCCWJ(tiny)
related work	Kaji [1]	**97.94**	**93.92**	98.10	N/A	N/A
	Kitagawa [2]	97.36	88.62	97.64	93.38	95.17
proposed	NN(char)	97.60	90.74	98.72	93.25	95.44
	NN(char+chartype)	97.82	91.33	98.76	94.04	95.92
	NN(scaling)	97.84	92.71	**98.89**	**94.78**	**96.58**

表 5　既成ソフトウェアとの F_1 の比較

	Method	KC	KNBC	BCCWJ	KC(tiny)	BCCWJ(tiny)
software	JUMAN	97.73	**96.26**	N/A	**97.64**	N/A
	MeCab	97.87	95.38	98.62	**97.64**	**98.47**
	Kytea	**98.20**	93.50	**99.05**	94.61	97.02
proposed	NN(scaling)	97.84	92.71	**98.89**	94.78	96.58

表 6　学習時間の比較 (分)

	Method	KC	KNBC	BCCWJ	KC(tiny)	BCCWJ(tiny)
related work	Kitagawa [2]	1,716	113	2,910	229	287
proposed	NN(char)	320	23	386	29	40
	NN(char+chartype)	426	33	516	43	60
	NN(scaling)	426	33	538	41	60

4.3 実験結果

評価実験では，比較手法として，鍛冶ら [1]，北川ら [2] の手法を用いた．ここで，鍛冶ら [1] の手法との比較は論文の値を参照したが，形態素解析手法であることから，単語分割が正しいかつ品詞タグが正しいものを正解とした．また，北川ら [2] の手法については，最も精度の高かった LSTM モデルを提案手法と同様に実装し評価した．各種ハイパーパラメータ k, d_c, d_p は北川ら [2] の設定した 5, 100, 10 を用いた．

提案手法については，文字 embedding のみで学習したモデル，文字と文字種 embedding で学習したモデル，文字と文字種 embedding で確率的文字種スケーリングを用いて学習したモデルの 3 つの手法について評価した．以下それぞれ NN(char), NN(char+chartype), NN(scaling) と表す．

4.3.1 既存手法との比較結果

既存手法と F_1 を比較した結果を表 4 に示す．結果から，KC, KNBC, BCCWJ のデータセットについては，鍛冶らの手法の方が F_1 が高くなったが，同じニューラルネットワークを用いた北川らの手法と比べて向上が見られた．KNBC のデータセットにおいて，鍛冶らの手法と F_1 に差が見れられのは，データセットが少ないため辞書により得られる特徴が強く影響したと考えられる．また，KNBC,KC(tiny),BCCWJ(tiny) の 3 つの小規模なデータセットについては，提案手法である確率的文字種スケーリングによる F_1 の向上がより顕著に確認できた．

4.3.2 既存ソフトウェアとの比較結果

また，既存の JUMAN, MeCab, KyTea の 3 つのソフトウェアと F_1 を比較した結果を表 5 に示す．JUMAN と MeCab については既に学習済みのモデルを用いているため，どのデータセットに対しても F_1 が高い結果となっている．KyTea に関しては，各データセットごとに学習したモデルを利用したが，KC(tiny) などの小規模のデータセットにおいては，提案手法の方が高い F_1 を得られた．

表 7　解析速度の比較 (文/秒)

	Method	Analysis Speed
software	JUMAN	2,100
	MeCab	29,000
	Kytea	3,200
related work	Kaji [1]	1,400
	Kitagawa [2]	12
proposed	NN(char)	8
	NN(char+chartype)	7
	NN(scaling)	7

4.3.3 ニューラルネットワークの学習時間の比較

提案手法と北川ら [2] の手法の学習時間の比較を表 6 に示す．ここで，各手法はエポック数を 100，バッチサイズを 100 とした場合の結果を示している．北川らの手法で用いている LSTM モデルでは，一つ前の中間層が次の中間層への入力となるため学習に多くの時間がかかっている．提案手法では，NN(char) と NN(char+chartype) で文字種 embedding の有無が学習時間に影響している．対して，NN(char+chartype) と確率的文字種スケーリングを適用した NN(scaling) との差はほとんど見られなかった．

4.3.4 解析速度の比較

解析速度の比較を表 7 に示す．提案手法については，解析速度の高速化は考慮していないので低い値となっている．時間計算量についても，北川らの手法が $\mathcal{O}(n)$ で実行できるのに対し，提案手法ではウィンドウサイズ k のとき，$\mathcal{O}(k^2 n)$ となってしまう．そのため，実用的な解析速度を得るために高速化や時間計算量について考慮する必要があると考えられる．

表 8　未知語カバー率の比較

Method		KC	KNBC	BCCWJ	KC(tiny)	BCCWJ(tiny)
proposed	NN(char)	97.07	88.32	98.09	91.72	93.41
	NN(char+chartype)	97.34	89.01	98.34	93.02	94.95
	NN(scaling)	97.37	91.16	98.36	93.58	95.27

4.3.5　未知語カバー率の比較

提案手法における未知語カバー率の比較を表 8 に示す．ここで，未知語とは訓練データ中に存在しない単語であり，未知語カバー率とは評価用データセット中の未知語全体に対する正しく分割された未知語の割合である．確率的文字種スケーリングを適用することで未知語カバー率においても向上が見られた．また表 4 と比較すると，未知後カバー率の向上が F_1 に大きく影響していることが分かった．

5. ま と め

本稿では，コーパスの規模に依存せず学習可能な単語分割手法について提案した．具体的には，入力文字をある確率に従って未知の文字として学習することで，未知の入力に対しても頑健なモデルの学習や過学習の回避を図った．評価実験の結果，既存のニューラルネットワークを用いた手法 [2] に対して提案手法では，F_1 が平均 94.43 から 96.00 へ向上したことにより有効性を確認した．しかし，特徴量エンジニアリングによる手法と比べて，精度の向上の必要性があると考えられる．今後の課題としては，解析速度の向上，品詞タグ付け，辞書情報を利用した学習方法の検討などが挙げられる．

文　献

[1] 鍛治伸裕, 喜連川優: "形態素解析における高速な単語ラティス生成", 人工知能学会論文誌, Vol.29, No.2, pp.268-276, 2014.

[2] 北川善彬, 小町守: "深層ニューラルネットワークを利用した日本語単語分割", 言語処理学会 第 22 回年次大会 発表論文集, pp.933–936, 2016.

[3] Kaji Nobuhiro and Kitsuregawa Masaru: *"Efficient Word Lattice Generation for Joint Word Segmentation and POS Tagging in Japanese."*, IJCNLP, pp.153-161, 2013.

[4] Morita Hajime, Kawahara Daisuke and Kurohashi Sadao: *"Morphological Analysis for Unsegmented Languages using Recurrent Neural Network Language Model"*, EMNLP, 2015.

[5] Neubig Graham, Nakata Yosuke and Mori Shinsuke: *"Pointwise prediction for robust, adaptable Japanese morphological analysis"*, Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2, pp.529–533, 2011.

[6] Chen Xinchi, Qiu Xipeng, Zhu Chenxi and Huang Xuanjing: *"Gated recursive neural network for Chinese word segmentation"*, Proceedings of Annual Meeting of the Association for Computational Linguistics. pendency parsing using two heterogeneous gated recursive neural networks. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, 2015.

[7] Zheng Xiaoqing, Chen Hanyang and Xu Tianyu: *"Deep Learning for Chinese Word Segmentation and POS Tagging."*, EMNLP, pp.647–657, 2013.

[8] Kurohashi Sadao and Nagao Makoto: *"Building a Japanese parsed corpus while improving the parsing system"*, Proceedings of The 1st International Conference on Language Resources & Evaluation, pp.719–724, 1998.

[9] Hashimoto Chikara, Kurohashi Sadao, Kawahara Daisuke, Shinzato Keiji and Nagata Masaaki: *"Construction of a Blog Corpus with Syntactic, Anaphoric, and Sentiment Annotations"*, Journal of Natural Language Processing, Vol.18, No.2, pp.175–201, 2011.

[10] Maekawa Kikuo, Yamazaki, Makoto, Ogiso Toshinobu, Maruyama Takehiko, Ogura Hideki, Kashino Wakako, Koiso Hanae, Yamaguchi Masaya, Tanaka Makiro and Den Yasuharu: *"Balanced corpus of contemporary written Japanese"*, Language Resources and Evaluation, Vol.48, No.2, pp.345–371, 2014.

[11] Tokui Seiya, Oono Kenta, Hido Shohei and Clayton Justin: *"Chainer: a Next-Generation Open Source Framework for Deep Learning"*, Proceedings of Workshop on Machine Learning Systems (LearningSys) in The Twenty-ninth Annual Conference on Neural Information Processing Systems (NIPS), 2015.