

系列ラベリングの多層化

東 藍[†] 松本 裕治[†]

[†] 奈良先端科学技術大学院大学情報科学研究科 〒 630-0192 奈良県生駒市高山町 8916-5

E-mail: †{ai-a,matsu}@is.naist.jp

あらまし 系列ラベリングには自然言語処理を始めとして幅広い応用分野がある。実用的なタスクにおいては、複数の系列ラベリングを段階的に適用する必要が生じる場合が多い。本発表では、各段階の系列ラベリングにおける素性に前段階の系列ラベリングの解析結果の周辺尤度を反映させると同時に、各段階の確率モデルの最適化において前段階のモデルパラメタを同時に最適化する手法を提案し、実用的なタスクにおける性能実験を紹介する。

キーワード 系列ラベリング, 結合学習, 構造学習

Multilayer Sequence Labeling

Ai AZUMA[†] and Yuji MATSUMOTO[†]

[†] Graduate School of Information Science, Nara Institute of Science and Technology

Takayama-cho 8916-5, Ikoma, Nara, 630-0192 Japan

E-mail: †{ai-a,matsu}@is.naist.jp

Abstract Sequence labeling has wide application areas such as natural language processing. In real world tasks, we often need to cascade multiple sequence labelers. We propose a method that combines features in each labeling stage with the corresponding marginal probabilities offered by preceding labelers. We also propose efficient algorithms that enable joint optimization of the model parameters of the current and preceding labelers. We also report experiments of cascaded part-of-speech tagging and chunking of English sentences and show the effectiveness of the proposed method.

Key words sequence labeling, sequential labeling, joint learning, structured prediction

1. ま え が き

あるインスタンスを離散的なカテゴリ(ラベル)に分類する問題に対して機械学習の手法が広く適用されている。しかしながら多くのタスクでは、互いに関連しあった複数のラベルを同時に決定する必要が生じる。このような問題を対象とする機械学習は特に構造学習と呼ばれる。系列ラベリングは構造学習の最も単純な部分クラスであり、与えられた入力に対して可能なすべてのラベル列候補集合からもっともらしいラベル列を推定する問題である。系列ラベリングは、機械学習の最も単純な部分クラスであるものの、多くの実用的タスクがこのクラスの問題としてモデル化される。加えて系列ラベリングは、より複雑で一般的な構造予測問題を考える上で価値のある洞察と足がかりをもたらすと思われる。多くのモデルが系列ラベリング問題をモデル化するために提案されている。たとえば、隠れマルコフモデル (Hidden Markov Models; HMM) や条件付確率場 (Conditional Random Fields; CRF)[1], 最大マージンマルコフネットワーク (Max-margin Markov Networks; M3N)[2] などが挙げられる。

自然言語処理や遺伝子系列解析、音声認識などにおける多くの実用問題が系列ラベリングでモデル化され、著しい成果を挙げている。

実際的なタスクにおいてはさらに、機械学習の手法によるラベル推定を複数階層的につなぎ合わせる必要がしばしば生じる。系列ラベリングも例外ではない。たとえば自然言語処理 (natural language processing; NLP) においては、与えられた文(単語列)に対して品詞ラベル付けを行い、さらに品詞ラベル付けの結果をもとにして固有表現抽出 (named entity recognition) や基本句の切り出し (base-phrase chunking) を行う必要が生じる。特に自然言語は、音素列、形態素列、固有表現や文節などの連続部分列などと、異なる抽象度のレベルからなる系列構造の階層を持つものとして自然に理解される。したがって、NLP の各種タスクも異なる抽象度のレベルを持つ系列の予測問題を階層的に適用する形式でモデル化されることが多い。

複数のラベル推定を階層的につなぎ合わせる問題に対して適用される手法には、ある種の性質を持つことが要求される。こ

の性質は2つに分類される。説明の便宜上、ある系列ラベリングを L_1 とし、 L_1 の出力（推定されたラベル列）に基づく別の系列ラベリングを L_2 と呼ぶことにする。要求される性質の一方の分類は、 L_2 が、 L_1 が出力する情報を可能な限り切り落とすことなく利用することであり、これを優れた前向きの情報伝達と呼ぶことにする。たとえば L_1 に用いられるモデルが確率モデルであるならば、 L_2 は L_1 の最尤出力系列だけでなく、次点の出力候補や各出力候補の確信度などを利用できる。要求される性質の他方の分類は後ろ向きの情報伝達である。 L_2 に対して作成された正解ラベル付きデータは、単に L_2 に対するモデルを最適化するためだけに使われるのではなく、 L_1 に対するモデルを構築する上でも何らかの形で反映されることが望ましい。

複数の系列ラベリングのシステムをつなぎ合わせてより複雑な解析システムを構築する場合、単純に各解析段階の 1-best 出力を次の解析の入力とする設計が多い。この設計は、既存の系列ラベリングのシステムの出力を単純に別の系列ラベリングのシステムの入力に直結させることで複雑な解析を対象とするシステムを組み上げることができ、したがって既存システムの再利用による工数の縮減という意味で構築が極めて容易である。一方でこのような単純な 1-best パイプラインによるシステムの構築は解析誤りの伝搬・拡大が顕著になるという明確な欠点を持つ。

1-best パイプラインによる解析誤りの伝搬・拡大をある程度改善することを目的として、 k -best を用いたパイプラインおよび前段階の出力を確率的に保持する手法が挙げられる。しかしながら、 k が増大すればするほど k -best 集合を効率的に列挙・保持することが困難になってくる。

k -best による前向きの情報伝達の改善を極端に推し進めたものとして、前段階のすべての出力候補に関する確信度付き情報を次段階の解析において利用する方法が挙げられる。ただし、系列ラベリングにおいてはすべての出力候補の数は膨大であるため、これらを効率的かつコンパクトな表現のまま取り扱うための工夫を必要とする。Finkel ら [3] では、前段階の推定に用いる確率モデルが出力する確信度に基づいて前段階の出力を複数サンプリングし、そこから得られる前段階の出力に関する素性を確率的に推定し、次段階の解析モデルに組み込んでいる。Bunescu [4] では、前段階の出力に関する素性に、その素性が発火する系列の部分構造（頂点や辺）の周辺尤度を重みとして掛け合わせた上で、次段階の解析における素性として用いている。 k -best パイプラインやこれらの手法は前向きの情報伝達を改善するには有意義ではあるが、一方で後ろ向きの情報伝達には一切寄与しない。

優れた前向きの情報伝達と後ろ向きの情報伝達の双方を実現するものとしては複数の系列ラベリングの同時学習が挙げられる。Sutton ら [5] は、ある入力系列の各時間スライスに対して複数のラベルを付与する問題に対する同時学習の手法を提案している。これにより、ある入力系列に対して、各時間スライスが完全に整列する複数の系列ラベリング問題を同時に学習・推定することが可能ではある。しかしながら、このモデル

は内部に循環的な依存関係を持つベイジアンネットワークであり、一般に exact inference を実現する効率的なアルゴリズムが知られていない。したがって、学習や推定の際に必要な inference において何らかの近似的手法が必要となる。また、本手法が想定している対象はある入力系列の各要素ごとに複数のラベルを同時付与する場合である。より一般に、系列毎に含まれるラベル数が異なるなどの事情からラベル列を時間スライスにうまく整列できない場合について、どのように同時学習可能なモデルを構築できるのかはまったく自明ではない。

本稿では、以上の背景を踏まえて、学習や推定において必要な inference を動的計画法によって効率的かつ厳密に行える利点を維持しつつ、前向きの情報伝搬と後ろ向きの情報伝搬の双方を一定程度実現し、また準マルコフモデルのようなより複雑な系列ラベリングを階層的に接続できる、良い妥協点に位置するモデルを提案する。本稿で提案するモデルおよび最適化アルゴリズムは、Bunescu [4] の考えを自然に発展させたものである。前段階の出力系列候補の隣接ラベル対などに関する部分的特徴を素性とし、この素性の値に、隣接ラベル対を出力する周辺尤度を掛け合わせたものを次段階のモデル素性に組み込む。これによって、前段階の出力候補集合すべてが尤度による重み付きで重ねあわされた形で表現されることになる。ここまでは Bunescu の提案にほぼ沿ったものである。本稿における提案の背後にある直観は以下のようなものである。前段階の素性を周辺化して次段階の解析の素性に組み込む場合、前段階の素性は周辺化という操作を通して前段階のモデルパラメタに間接的に依存した変数とみなすことができる。したがって、次段階のモデルを最適化するにあたって、次段階の直接のモデルパラメタを変化させるだけではなく、前段階の出力に関する素性を周辺化したものも変数とみなして変化させることでより積極的に次段階のモデルを改善できる、というものである。前段階の素性を周辺化したものを変化させるということは、結局、前段階のモデルパラメタを間接的に変化させることになる。これは結局、次段階のモデルの最適化において前段階のモデルを同時に間接的に改善することとなる。

本稿の構成について述べる。次節では、提示するモデルの厳密な定式化について述べる。3. 節では、2. 節で定式化したモデルの最適化手順について述べる。4. 節では、提案するモデルの有効性を検証するため、実際のタスクにおいて行った評価実験について述べる。最後に、5. 節で本稿の結論について述べる。

2. 定式化

本節では提案するモデルの形式的な表記を導入する。以下、簡単のため、最も単純な場合についてのみ議論する。すなわち、ある系列ラベリングの段階 L_1 と、 L_1 に依存する別の系列ラベリングの段階 L_2 の、これら2つだけがある場合について議論する。 L_1 はある入力 x に対してもっともらしい系列を予測するものである。また L_2 は、入力 x に加えて L_1 の出力にも依存してもっともらしい系列を予測するものである。ここで入力 x についてはその構造などについて特に仮定を置かない。一般に系列ラベリングと言う場合には x は L_1, L_2 の出力と同じ

く離散ラベルの列が想定される場合が多いが、本稿では特にこれに限らない。入力 \mathbf{x} をどうモデルに取り込むかは素性関数の設計に一義的によるものとし、後は L_1 および L_2 の出力候補集合が \mathbf{x} に条件付けられているだけとみなす。

まず初めに L_1 に対する確率モデルを定式化する。 L_1 に対する確率モデルそれ自体は通常の系列ラベリングのそれと同等である。入力 \mathbf{x} に対して有向非循環グラフ (directed acyclic graph; DAG) $G_1 = (V_1, E_1)$ を考える。ある DAG G に対して、ソースを入次数が零の頂点と、シンクを出次数が零の頂点と各々定義する。また、ある DAG G において、ソースとなっているある頂点を始点、シンクとなっているある頂点を終点とする有向経路をサクセスフルな経路と呼ぶことにする。簡単のために、DAG のある経路を表す表記が与えられたとき、それをその経路上にあるすべての辺の集合を表す表記としても取り扱うようにする。 G_1 のすべてのサクセスフルな経路の集合を \mathbf{Y}_1 と置く。 L_1 の出力候補集合は \mathbf{Y}_1 となる。 L_1 のモデル化に用いられる素性関数を $f_{\langle 1, k_1, e_1, \mathbf{x} \rangle} \in \mathbb{R}$ ($k_1 \in \mathcal{K}_1, e_1 \in E_1$) と書く。ここで \mathcal{K}_1 は L_1 のモデル化に使用する素性タイプの添え字集合である。この、 L_1 のモデル化に用いられる種類の素性を L_1 に対する入力素性と呼ぶことにする。この命名は、後に L_1 に対して定義する別の種類の素性群と区別するためのものである。頂点 V_1 上にも素性関数を定義できるが、頂点上にも素性関数を定義した場合への定式化の拡張は極めて平易であり、本稿では簡単のため省略する。以降では、ある記号が下付き添え字を持つ場合に、同じ記号のボールド体で、かつ、いくつかの添え字が省略された表記が、省略された添え字をわたる集合を表すものとする。たとえば、 $f_{\langle 1, e_1, \mathbf{x} \rangle} \stackrel{\text{def}}{=} \{f_{\langle 1, k_1, e_1, \mathbf{x} \rangle}\}_{k_1 \in \mathcal{K}_1}$ 、 $\mathbf{f}_{\langle 1, k_1, \mathbf{x} \rangle} \stackrel{\text{def}}{=} \{f_{\langle 1, k_1, e_1, \mathbf{x} \rangle}\}_{e_1 \in E_1}$ 、 $\mathbf{f}_{\langle 1, \mathbf{x} \rangle} \stackrel{\text{def}}{=} \{f_{\langle 1, k_1, e_1, \mathbf{x} \rangle}\}_{k_1 \in \mathcal{K}_1, e_1 \in E_1}$ などとする。 L_1 に対する確率モデルは対数線形モデル、すなわち、

$$P_1(\mathbf{y}_1 | \mathbf{x}; \boldsymbol{\theta}_1) \stackrel{\text{def}}{=} \frac{1}{Z_1(\mathbf{x}; \boldsymbol{\theta}_1)} \exp(\boldsymbol{\theta}_1 \cdot \mathbf{F}_{\langle 1, \mathbf{y}_1, \mathbf{x} \rangle}) \quad (1)$$

$(\mathbf{y}_1 \in \mathbf{Y}_1)$

とする。ここで、 $\theta_{\langle 1, k_1 \rangle} \in \mathbb{R}$ ($k_1 \in \mathcal{K}_1$) は添え字 k_1 の素性タイプに対するパラメタであり、また $F_{\langle 1, k_1, \mathbf{y}_1, \mathbf{x} \rangle} \stackrel{\text{def}}{=} \sum_{e_1 \in \mathbf{Y}_1} f_{\langle 1, k_1, e_1, \mathbf{x} \rangle}$ とする。ドットで表した演算子 (\cdot) は、演算子の両オペランドで共通して省略されている下付き添え字に関する内積を表すものとする。たとえば、式 (1) のドット演算子は $\boldsymbol{\theta}_1 \cdot \mathbf{F}_{\langle 1, \mathbf{y}_1, \mathbf{x} \rangle} \stackrel{\text{def}}{=} \sum_{k_1 \in \mathcal{K}_1} \theta_{\langle 1, k_1 \rangle} F_{\langle 1, k_1, \mathbf{y}_1, \mathbf{x} \rangle}$ を意味する。 Z_1 は P_1 に対する分配関数であり、以下で定義される。

$$Z_1(\mathbf{x}; \boldsymbol{\theta}_1) \stackrel{\text{def}}{=} \sum_{\mathbf{y}_1 \in \mathbf{Y}_1} \exp(\boldsymbol{\theta}_1 \cdot \mathbf{F}_{\langle 1, \mathbf{y}_1, \mathbf{x} \rangle}) \quad (2)$$

上記の定式化は有向・無向双方の直鎖グラフィカルモデルを含んだものであることに注意されたい。直鎖グラフィカルモデルは系列ラベリングに対する最も典型的なモデルの 1 つであり、HMM や直鎖の CRF などを含む。すなわち、 V_1 の要素を時間スライスに整列させ、ある時間スライスに並んでいるノード群をその時間に対して可能なラベル割り当てと対応させ、さらに隣接した時間スライスにある頂点の対を有向辺で結

ぶことで、通常の直鎖グラフィカルモデルと同等な表現を得る。このように得られた G_1 は、ちょうど直鎖グラフィカルモデルの各状態における可能なラベル割り当てを全て展開した表現となる。このような配置により、本稿における表記 \mathbf{Y}_1 と、直鎖グラフィカルモデルにおいて可能なすべてのラベルの結合割り当ての集合との間に厳密な 1 対 1 関係を見出すことができる。我々は本稿におけるような DAG に基づく表記をあえて採用する。これは、第一に、我々が必要とするモデルやアルゴリズムの記述に適しているからであり、第二に、この表記により準マルコフモデルのような各状態が任意の滞留時間を持つことを許す場合が自然に包含されるからであり、第三に、この有向グラフによる表記を、ある種の有向超グラフによる表記に置き換えることにより Cocke–Kasami–Younger アルゴリズムなどによって導出される木の集合に対するモデルに自然に拡張できるからである。

次に L_2 に対する確率モデルを定式化する。 L_1 と同様、入力 \mathbf{x} に条件付けられた DAG $G_2 = (V_2, E_2)$ を考える。また、 G_2 において可能なすべてのサクセスフルな経路の集合を \mathbf{Y}_2 とおく。 L_2 の出力候補集合は \mathbf{Y}_2 となる。

L_2 に対して定義する確率モデルを P_2 とおく。 P_2 において使用する素性の形式が本稿において重要となる部分である。 L_1 と同様、 L_2 においても E_2 上に素性関数を定義する。ある辺 $e_2 \in E_2$ 上の素性は、入力 \mathbf{x} の情報に加えて、 L_1 のある出力候補系列の局所的な特徴を L_1 のすべての出力候補にわたって P_1 による確信度付きで重ね合わせて用いることができるとする。 L_1 のある出力候補系列の局所的な特徴を定式化するため、まず L_1 の出力素性なるものを $h_{\langle 1, k'_1, e_1 \rangle} \in \mathbb{R}$ ($k'_1 \in \mathcal{K}'_1, e_1 \in E_1$) という形式で定義する。ここで、 \mathcal{K}'_1 は L_1 の出力素性のタイプについての添え字集合である。これらの出力素性はすべて、 L_2 のモデル化に用いられる素性に組み込まれる前に、 L_1 のすべての出力候補系列にわたって P_1 による確信度付きで総和される。より形式的には、 L_1 のすべての出力素性は以下で定義する周辺化出力素性なる形式で P_2 の素性に組み込まれるものとする。 L_1 の各出力素性 $h_{\langle 1, k'_1, e_1 \rangle}$ に対して、対応する周辺化出力素性 $\bar{h}_{\langle 1, k'_1, e_1 \rangle}(\boldsymbol{\theta}_1)$ は以下で定義される。

$$\bar{h}_{\langle 1, k'_1, e_1 \rangle}(\boldsymbol{\theta}_1) \stackrel{\text{def}}{=} h_{\langle 1, k'_1, e_1 \rangle} P_1(e_1 | \mathbf{x}; \boldsymbol{\theta}_1) \quad (3)$$

$(k'_1 \in \mathcal{K}'_1, e_1 \in E_1)$.

ただし、

$$P_1(e_1 | \mathbf{x}; \boldsymbol{\theta}_1) \stackrel{\text{def}}{=} \sum_{\mathbf{y}_1 \sim e_1} P_1(\mathbf{y}_1 | \mathbf{x}; \boldsymbol{\theta}_1) \quad (4)$$

$$= \sum_{\mathbf{y}_1 \in \mathbf{Y}_1} \delta_{e_1 \in \mathbf{y}_1} P_1(\mathbf{y}_1 | \mathbf{x}; \boldsymbol{\theta}_1)$$

$(e_1 \in E_1)$.

ここで、 $\sum_{\mathbf{y}_1 \sim e_1}$ なる表記は辺 $e_1 \in E_1$ が現れる系列に関してのみ和を取ることを表す。すなわち、集合 $\{\mathbf{y}_1 \in \mathbf{Y}_1 \mid e_1 \in \mathbf{y}_1\}$ をわたる和である。また、 δ_P は叙述 P に対する指示関数である。 $e_2 \in E_2$ 上に定義される L_2 の入力素性は、 \mathbf{x} に関する情

報, L_1 の周辺化出力素性, ならびに e_2 の特徴を任意に組み合わせることで定義できるものとする. すなわち, L_2 における, 辺 $e_2 \in E_2$ 上の入力素性は次の形式を持つものとする.

$$f_{\langle 2, k_2, e_2, \mathbf{x} \rangle}(\bar{\mathbf{h}}_1(\boldsymbol{\theta}_1)) \in \mathbb{R} \quad (k_2 \in \mathcal{K}_2). \quad (5)$$

ここで, \mathcal{K}_2 は L_2 の入力素性のタイプに関する添え字集合である. 後に述べる最適化の手続きが実行可能になるよう, L_2 の任意の入力素性が L_1 のすべての周辺化出力素性に関して滑らかであるように定義されるという条件をおく. すなわち, $\frac{\partial f_{\langle 2, k_2, e_2, \mathbf{x} \rangle}}{\partial h_{\langle 1, k'_1, e_1 \rangle}}$ が $\forall k'_1, e_1, k_2, e_2$ について常に存在することを仮定する. たとえば, 単一の周辺化出力素性のみ, 複数の周辺化出力素性の和や積, 全域で微分可能な関数 (たとえばシグモイド関数) をそれらに適用したもの, これらと e_2 に関する叙述の指示関数との積, などから定義される入力素性はすべて上記の微分の存在に関する仮定を満たす. 与えられた入力素性 $f_{\langle 2, \mathbf{x} \rangle}(\bar{\mathbf{h}}_1(\boldsymbol{\theta}_1))$ と対応するパラメタ $\theta_{\langle 2, k_2 \rangle} \in \mathbb{R}$ ($k_2 \in \mathcal{K}_2$) に対して, L_2 に対する確率モデルを以下のように定義する.

$$P_2(\mathbf{y}_2 | \mathbf{x}; \boldsymbol{\theta}_1, \boldsymbol{\theta}_2) \stackrel{\text{def}}{=} \frac{1}{Z_2(\mathbf{x}; \boldsymbol{\theta}_1, \boldsymbol{\theta}_2)} \exp(\boldsymbol{\theta}_2 \cdot \mathbf{F}_{\langle 2, \mathbf{y}_2, \mathbf{x} \rangle}(\bar{\mathbf{h}}_1(\boldsymbol{\theta}_1))) \quad (6)$$

$(\mathbf{y}_2 \in \mathbf{Y}_2).$

ただし $F_{\langle 2, k_2, \mathbf{y}_2, \mathbf{x} \rangle}(\bar{\mathbf{h}}_1(\boldsymbol{\theta}_1)) \stackrel{\text{def}}{=}} \sum_{e_2 \in \mathbf{Y}_2} f_{\langle 2, k_2, e_2, \mathbf{x} \rangle}(\bar{\mathbf{h}}_1(\boldsymbol{\theta}_1))$ とする. また Z_2 は P_2 に対する分配関数であり次のように定義される.

$$Z_2(\mathbf{x}; \boldsymbol{\theta}_1, \boldsymbol{\theta}_2) \stackrel{\text{def}}{=} \sum_{\mathbf{y}_2 \in \mathbf{Y}_2} \exp(\boldsymbol{\theta}_2 \cdot \mathbf{F}_{\langle 2, \mathbf{y}_2, \mathbf{x} \rangle}(\bar{\mathbf{h}}_1(\boldsymbol{\theta}_1))). \quad (7)$$

本稿において最も重要な点の1つが P_2 の定義 (6) に現れている. P_2 は, 直接のパラメタ $\boldsymbol{\theta}_2$ の関数としてみなされるだけでなく $\boldsymbol{\theta}_1$ の関数としてもみなされる. これは, L_1 のモデルパラメタ $\boldsymbol{\theta}_1$ が中間的な変数 $\bar{\mathbf{h}}_1$ を通じて P_2 に間接的に影響を与えていることを反映している. したがって P_2 に対する最適化は, 直接のモデルパラメタ $\boldsymbol{\theta}_2$ だけでなく, 間接的なモデルパラメタ $\boldsymbol{\theta}_1$ の決定にも影響を与えるのである.

ここで, 仮に L_1 の出力がただ一つの出力に縮退したとすると, すなわち $P_1(\mathbf{y}_1 | \mathbf{x}) = \delta_{\mathbf{y}_1 = \hat{\mathbf{y}}_1}$ とおく. すると, 上記の定式化は, 通常みられるような独立した系列ラベリングの処理を階層的に接続するモデルの定式化に還元される. ただし, 独立した系列ラベリングの処理を階層的に接続する場合には, L_1 のただ一つの出力についてその全体的な情報, すなわち大域的な素性を L_2 の解析に用いることが可能になるが, 上記で提案したモデルでは L_1 の出力の局所的な特徴しか L_2 の解析に用いることができないという制約を持つことになる.

3. 最適化アルゴリズム

本節では, 前節で提案したモデルに対するパラメタ最適化の手続きについて記述する. パラメタ最適

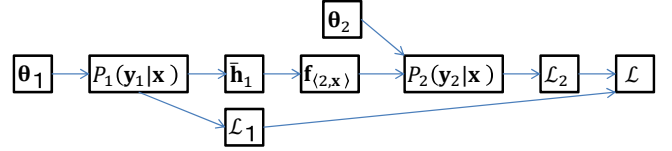


図 1 提案したモデルの計算グラフ

化に用いることのできる正解ラベル付きデータを $\mathcal{D} = \{\langle \hat{\mathbf{x}}, \langle G_1, \hat{\mathbf{y}}_1 \rangle, \langle G_2, \hat{\mathbf{y}}_2 \rangle \rangle_m\}_{m=1,2,\dots,M}$ とおく. ここで各 $\langle G_1, \hat{\mathbf{y}}_1 \rangle$ は, L_1 における正解候補系列集合を表現する DAG と正解系列の対であり, $\langle G_2, \hat{\mathbf{y}}_2 \rangle$ も同様である. この \mathcal{D} に対して, L_1 および L_2 における条件付対数尤度関数を各々定義する. すなわち,

$$\mathcal{L}_1(\boldsymbol{\theta}_1; \mathcal{D}) \stackrel{\text{def}}{=} \sum_{\langle \hat{\mathbf{x}}, \hat{\mathbf{y}}_1 \rangle \in \mathcal{D}} \log(P_1(\hat{\mathbf{y}}_1 | \hat{\mathbf{x}}; \boldsymbol{\theta}_1)) - \frac{|\boldsymbol{\theta}_1|^2}{2\sigma_1^2} \quad (8)$$

と

$$\mathcal{L}_2(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2; \mathcal{D}) \stackrel{\text{def}}{=} \sum_{\langle \hat{\mathbf{x}}, \hat{\mathbf{y}}_2 \rangle \in \mathcal{D}} \log(P_2(\hat{\mathbf{y}}_2 | \hat{\mathbf{x}}; \boldsymbol{\theta}_1, \boldsymbol{\theta}_2)) - \frac{|\boldsymbol{\theta}_2|^2}{2\sigma_2^2}. \quad (9)$$

である. σ_1^2, σ_2^2 は各々パラメタの事前分布の分散である. ここでは簡単のため, 事前分布として, 平均 0 で分散が各々の系列ラベリングの全パラメタで共通のガウス分布とした. さらに次の同時条件付対数尤度関数

$$\mathcal{L}(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2; \mathcal{D}) \stackrel{\text{def}}{=} \mathcal{L}_1 + \mathcal{L}_2. \quad (10)$$

を定義し, 本稿ではこれを P_1 と P_2 を同時最適化するための目標関数とする. したがって, 最適化の結果学習されるパラメタ値はこの目標関数 \mathcal{L} を (局所的に) 最大化するものである.

本稿では勾配法に基づく最適化手順を仮定する. 最適化においては, パラメタ空間において目標関数を増大させる方向を逐次探索し, その方向に沿ってパラメタ値を更新することを繰り返す. この手順は, 最急勾配法や共役勾配法を始めとする既存の多くの最適化ルーチンを用いれば, 与えられたパラメタ値における目標関数の値とパラメタ勾配を与えることを繰り返すだけで達成される. したがって, 最適化問題は与えられたパラメタ値に対して目標関数の値とパラメタ勾配を計算する問題に帰着される.

目標関数とパラメタ勾配を計算するアルゴリズムの詳細に立ち入る前に, これまでに定義した目標関数および変数間の関数関係について注記しておく. 図 1 を示す. 本図は, パラメタ, 入力・出力素性関数, モデル, そして目標関数の間の関数関係を示したものである. 図中の各矢印は, 矢印の先にある変数とその矢印の根元にある変数によって直接に定義されていることを示したものである. 目標関数の値は図中の矢印の順にしたがって計算される. 他方, パラメタ勾配は矢印の逆順に沿って順に計算される. 図 1 に示された関数関係は, 図中の変数間におけるある形式の微分の連鎖則 (合成関数に対する微分法) の存在を保証する. この連鎖則を繰り返し用いることで, 勾配

計算を分割統治的な計算方法に分解することが可能になる．これらの、目標関数を計算するための順方向、およびパラメタ勾配を計算するための逆方向の段階的な計算は、各々、多層神経回路における前向き伝搬、および後ろ向き伝搬（誤差逆伝搬）に類似したものとすることもできる．

Algorithm 1 は、本稿で提案したモデルに対する勾配法による最適化の全体像を表したものである．まず初めに、与えられたパラメタ θ_1, θ_2 に対して目標関数の値を計算する手順について述べる．この手順は Algorithm 1 中の 2 行目から 4 行目に対応する．周辺化出力素性 $\bar{\mathbf{h}}_{(1,\mathbf{x})}$ の値は定義 (3) から計算される．この計算は単純な素性の周辺化であるため、通常の前向き後ろ向きアルゴリズムによって効率的に求めることができる．次に L_2 の各入力素性 $f_{(2,k_2,e_2,\mathbf{x})}$ ($k_2 \in \mathcal{K}_2, e_2 \in E_2$) の値を計算する．これらの具体的な計算方法は各入力素性の定義に依るが、いずれにせよ L_1 の周辺化出力素性 $\bar{\mathbf{h}}_{(1,\mathbf{x})}$ の値が計算されればただちに求まる．最後に、与えられたパラメタ値とすでに計算された L_1, L_2 の各入力素性の値から \mathcal{L}_1 および \mathcal{L}_2 はただちに求められる．これらは条件付確率場における通常の対数尤度計算となんら変わらない．したがって、目標関数 \mathcal{L} の値もただちに求まる．

次にパラメタ勾配

$$\frac{\partial \mathcal{L}}{\partial \theta_1} = \frac{\partial \mathcal{L}_1}{\partial \theta_1} + \frac{\partial \mathcal{L}_2}{\partial \theta_1}, \quad \frac{\partial \mathcal{L}}{\partial \theta_2} = \frac{\partial \mathcal{L}_2}{\partial \theta_2} \quad (11)$$

の計算アルゴリズムについて述べる．Algorithm 1 中の 5 行目から 7 行目が勾配計算に対応する．式 (11) 中の項、 $\frac{\partial \mathcal{L}_1}{\partial \theta_1}$ および $\frac{\partial \mathcal{L}_2}{\partial \theta_2}$ の計算は容易である．なぜなら、これらは通常の条件付確率場の最適化において現れる形式、つまり素性の経験期待値とモデル期待値の差という形式となるからである．

$$\begin{aligned} \frac{\partial \mathcal{L}_1}{\partial \theta_1} &= \tilde{E}[\mathbf{F}_{(1,\mathbf{y}_1,\mathbf{x})}] - E_{P_1}[\mathbf{F}_{(1,\mathbf{y}_1,\mathbf{x})}] - \frac{|\theta_1|}{\sigma_1^2}, \\ \frac{\partial \mathcal{L}_2}{\partial \theta_2} &= \tilde{E}[\mathbf{F}_{(2,\mathbf{y}_2,\mathbf{x})}] - E_{P_2}[\mathbf{F}_{(2,\mathbf{y}_2,\mathbf{x})}] - \frac{|\theta_2|}{\sigma_2^2}. \end{aligned} \quad (12)$$

これらは G_1, G_2 各々に対する通常の前向き後ろ向きアルゴリズムを用いることで効率的に計算できる．図 1 に示した関係から導出される合成関数の微分則を用いることで、式 (11) における残りの項 $\frac{\partial \mathcal{L}_2}{\partial \theta_1}$ は次のように分解される．

$$\frac{\partial \mathcal{L}_2}{\partial \theta_1} = \frac{\partial \mathcal{L}_2}{\partial \mathbf{f}_{(2,\mathbf{x})}} \cdot \frac{\partial \mathbf{f}_{(2,\mathbf{x})}}{\partial \theta_1} = \frac{\partial \mathcal{L}_2}{\partial \mathbf{f}_{(2,\mathbf{x})}} \cdot \frac{\partial \mathbf{f}_{(2,\mathbf{x})}}{\partial \mathbf{h}_1} \cdot \frac{\partial \bar{\mathbf{h}}_1}{\partial \theta_1}. \quad (13)$$

ここで、ライブニッツの記法は分子あるいは分母において省略された添え字集合に関するヤコビアンを表すものとする．たとえば次のとおりである．

$$\frac{\partial \mathbf{f}_{(2,\mathbf{x})}}{\partial \mathbf{h}_1} \stackrel{\text{def}}{=} \left\{ \frac{\partial f_{(2,k_2,e_2,\mathbf{x})}}{\partial h_{(1,k'_1,e_1)}} \right\}_{k_2 \in \mathcal{K}_2, e_2 \in E_2, k'_1 \in \mathcal{K}'_1, e_1 \in E_1} \quad (14)$$

またドット演算子は、先に定義した通り、その両オペランドで共通して省略された添え字集合に関する内積を表すものとする．たとえば以下のとおりである．

$$\begin{aligned} &\frac{\partial \mathcal{L}_2}{\partial \mathbf{f}_{(2,\mathbf{x})}} \cdot \frac{\partial \mathbf{f}_{(2,\mathbf{x})}}{\partial \mathbf{h}_1} \\ &= \sum_{k_2 \in \mathcal{K}_2, e_2 \in E_2} \frac{\partial \mathcal{L}_2}{\partial f_{(2,k_2,e_2,\mathbf{x})}} \cdot \frac{\partial f_{(2,k_2,e_2,\mathbf{x})}}{\partial h_1}. \end{aligned} \quad (15)$$

以下、式 (13) の最右辺の各因子の変形について順に述べる． $\frac{\partial f_{(2,k_2,e_2,\mathbf{x})}}{\partial f_{(2,k_2,e_2,\mathbf{x})}} = \delta_{k_2=k_2} \delta_{e_2=e_2}$ という関係に注意すると、式 (13) 最右辺の最初の因子 $\frac{\partial \mathcal{L}_2}{\partial \mathbf{f}_{(2,\mathbf{x})}}$ の各要素は次のように変形される．

$$\begin{aligned} &\frac{\partial \mathcal{L}_2}{\partial f_{(2,k_2,e_2,\mathbf{x})}} \\ &= \theta_{(2,k_2)} \sum_{(\hat{\mathbf{x}}, \hat{\mathbf{y}}_2) \in \mathcal{D}} (\delta_{e_2 \in \hat{\mathbf{y}}_2} - P_2(e_2 | \hat{\mathbf{x}}; \theta_1, \theta_2)). \end{aligned} \quad (16)$$

上式に現れる $P_2(e_2 | \hat{\mathbf{x}}; \theta_1, \theta_2)$ は e_2 についての周辺化確率にほかならず、したがって式 (12) を計算するにあたって実行した前向き後ろ向きアルゴリズムの副産物としてただちにその値が得られる．この式 (16) を各 $k_2 \in \mathcal{K}_2, e_2 \in E_2$ に対して計算することで第一の因子の全要素の値が求まる．

前節で述べたとおり、式 (13) 最右辺の第二因子 $\frac{\partial \mathbf{f}_{(2,\mathbf{x})}}{\partial \mathbf{h}_1}$ の値はあらゆる θ_1 について常に存在し計算できると仮定していた．この計算の具体的な内容はやはり L_2 の各入力素性の定義によるが、いずれにせよ、目標関数の計算においてすでに求めていた $\bar{\mathbf{h}}_1$ の値からこの偏微分係数はただちに求めることができる．この第二の因子が計算できれば、先に計算した第一の因子 $\frac{\partial \mathcal{L}_2}{\partial \mathbf{f}_{(2,\mathbf{x})}}$ との内積を取ることで $\frac{\partial \mathcal{L}_2}{\partial \theta_1} = \frac{\partial \mathcal{L}_2}{\partial \mathbf{f}_{(2,\mathbf{x})}} \cdot \frac{\partial \mathbf{f}_{(2,\mathbf{x})}}{\partial \mathbf{h}_1}$ をただちに得る．

$\bar{\mathbf{h}}_1$ の定義である式 (3) から、式 (13) 最右辺の第三因子 $\frac{\partial \bar{\mathbf{h}}_1}{\partial \theta_1}$ の各要素は以下のように変形される．

$$\begin{aligned} &\frac{\partial \bar{h}_{(1,k'_1,e_1)}}{\partial \theta_{(1,k_1)}} \\ &= h_{(1,k'_1,e_1)} \text{Cov}_{P_1(\mathbf{y}_1|\mathbf{x})} [\delta_{e_1 \in \mathbf{y}_1}, F_{(1,k_1,\mathbf{y}_1,\mathbf{x})}]. \end{aligned} \quad (17)$$

共分散 (17) を計算する効率的な動的計画法は存在する（ただしここではその詳細には立ち入らない．なぜならそれは後に示すアルゴリズムから容易に類推されるからである）．したがって、その動的計画法を各 $k'_1 \in \mathcal{K}'_1, e_1 \in E_1, k_1 \in \mathcal{K}_1$ について実行すれば第三因子の全要素を計算できる．しかしながら、ヤコビアン $\frac{\partial \bar{\mathbf{h}}_1}{\partial \theta_1}$ の大きさは $|\mathcal{K}'_1| \times |E_1| \times |\mathcal{K}_1|$ であり、これは実際のタスクにおいては極めて大きくなりうる．したがってこのヤコビアン各要素を直接計算することは避けなければならない．この愚直な計算の代わりに、先に計算した $\frac{\partial \mathcal{L}_2}{\partial \mathbf{h}_1}$ を用いることで、式 (13) と (17) から

$$\begin{aligned} &\frac{\partial \mathcal{L}_2}{\partial \theta_1} = \frac{\partial \mathcal{L}_2}{\partial \bar{\mathbf{h}}_1} \cdot \frac{\partial \bar{\mathbf{h}}_1}{\partial \theta_1} \\ &= E_{P_1(\mathbf{y}_1|\mathbf{x})} [H'_{(1,\mathbf{y}_1)} \mathbf{F}_{(1,\mathbf{y}_1,\mathbf{x})}] \\ &\quad - E_{P_1(\mathbf{y}_1|\mathbf{x})} [H'_{(1,\mathbf{y}_1)}] E_{P_1(\mathbf{y}_1|\mathbf{x})} [\mathbf{F}_{(1,\mathbf{y}_1,\mathbf{x})}] \end{aligned} \quad (18)$$

と変形できる．ただし、 $H'_{(1,\mathbf{y}_1)} \stackrel{\text{def}}{=} \sum_{e_1 \in \mathbf{y}_1} \frac{\partial \mathcal{L}_2}{\partial h_{(1,e_1)}} \cdot \mathbf{h}_{(1,e_1)}$ である．言い換えれば $\frac{\partial \mathcal{L}_2}{\partial \theta_{(1,k_1)}}$ は、 L_1 の k_1 番目の入力素性と、各 $e_1 \in E_1$ 上に定義された仮想的な素性 $h'_{(1,e_1)} \stackrel{\text{def}}{=} \frac{\partial \mathcal{L}_2}{\partial h_{(1,e_1)}} \cdot \mathbf{h}_{(1,e_1)}$ との P_1 に関する共分散に等しい．この事実に基づいて、まず仮想的な素性関数 $h'_{(1,e_1)}$ をあらかじめ各 $e_1 \in E_1$ に対して計算しておいてから式 (18) の共分散の計算を行うことで、求める偏微分係数を効率的に得ることができる．

Algorithm 1 勾配法に基づくモデルパラメタ最適化

Input: θ_1, θ_2 **Output:** $\operatorname{argmax}_{(\theta_1, \theta_2)} \mathcal{L}$

- 1: **while** θ_1 or θ_2 changes significantly **do**
 - 2: calculate Z_1 by (2), $\bar{\mathbf{h}}_1$ by (3) with the forward-backward algorithm (F-B) on G_1 , and then \mathcal{L}_1 by (8)
 - 3: calculate $\mathbf{f}_{(2, \mathbf{x})}$ according to their definitions
 - 4: calculate Z_2 by (7) with the F-B on G_2 , and then \mathcal{L}_2 by (9) and \mathcal{L} by (10)
 - 5: calculate $\frac{\partial \mathcal{L}_1}{\partial \theta_1}$ and $\frac{\partial \mathcal{L}_2}{\partial \theta_2}$ by (12) with the F-B on G_1 and G_2 , respectively
 - 6: calculate $\frac{\partial \mathcal{L}}{\partial \mathbf{f}_{(1, \mathbf{x})}}$ by (16) with the F-B on G_2 , $\frac{\partial \mathbf{f}_{(1, \mathbf{x})}}{\partial \mathbf{h}_1}$, and then $\frac{\partial \mathcal{L}}{\partial \mathbf{h}_1} = \frac{\partial \mathcal{L}}{\partial \mathbf{f}_{(1, \mathbf{x})}} \cdot \frac{\partial \mathbf{f}_{(1, \mathbf{x})}}{\partial \mathbf{h}_1}$
 - 7: calculate $\frac{\partial \mathcal{L}}{\partial \theta_1}$ by (18) with Algorithm 2
 - 8: $(\theta_1, \theta_2) \leftarrow \text{update-parameters}(\theta_1, \theta_2, \mathcal{L}, \frac{\partial \mathcal{L}}{\partial \theta_1}, \frac{\partial \mathcal{L}}{\partial \theta_2})$
 - 9: **end while**
-

パラメタ勾配を求める上で最終的に残った問題は、式 (18) 最右辺の第 1 項を効率的に計算する方法を導出することである。式 (18) 最右辺の第 2 項は素性の単純な周辺化のみからなるため、通常の前向き後ろ向きアルゴリズムを用いてただちに求めることができる。この第 1 項を計算するアルゴリズムには 2 つの導出方法があり、その両方について簡単に言及しておく。

1 つは期待値半環 (expectation semi-ring) [6] と呼ばれる代数構造に対して一般化した前向き後ろ向きアルゴリズムを用いることで導出される。先に定義した仮想的な素性 $h'_{(1, e_1)}$ に関する期待値半環上に対して定義される前向き後ろ向きアルゴリズムの変種を実行し、この期待値半環がなすポテンシャル上で素性ベクトル $\mathbf{f}_{(1, e_1, \mathbf{x})}$ を各辺で周辺化し、それらの和を取ることによって、 $h'_{(1, e_1)}$ に関する期待値半環上の $\mathbf{f}_{(1, e_1, \mathbf{x})}$ の期待値、すなわち式 (18) 最右辺の第 1 項を得る。素性ベクトル $\mathbf{f}_{(1, e_1, \mathbf{x})}$ が、 $h'_{(1, e_1)}$ に関する期待値半環に対して P -module となることからこの期待値計算が可能なが保証される。^(注1)

もう 1 つの導出方法は、 $E_{P_1} [\mathbf{F}_{(1, y_1, \mathbf{x})}]$ を計算する前向き後ろ向きアルゴリズムに自動微分法 [7], [8] を適用するものである。これは $\left. \frac{\partial}{\partial \lambda} E_{P_1} [\mathbf{F}_{(1, y_1, \mathbf{x})}] \right|_{\lambda=0}$ が式 (18) 最右辺の第 1 項に一致する、という事実を利用するものである。ここで、 $\lambda \in \mathbb{R}$ はダミーのパラメタであり、 P'_1 は $P'_1(y_1 | \mathbf{x}) \stackrel{\text{def}}{=} \frac{1}{Z_1} \exp(\theta_1 \cdot \mathbf{F}_{(1, y_1, \mathbf{x})} + \lambda H'_{(1, y_1)})$ と定義される。 $E_{P'_1} [\mathbf{F}_{(1, y_1, \mathbf{x})}] \Big|_{\lambda=0}$ の値を計算する前向き後ろ向きアルゴリズムは容易に導出できる。自動微分法は、この前向き後ろ向きアルゴリズムを、 λ に関して微分したものを $\lambda = 0$ で評価した値を計算するアルゴリズムに変換するものである。このアルゴリズム・アルゴリズム間の変換は機械的に実行することができ、これは前向き後ろ向きアルゴリズムにおける λ のすべての出現を双対数 $\lambda + \varepsilon$ に機械的に置き換えることで得られる。双対数とは複素数の変種であり、 $\varepsilon^2 = 0$ なる性質を持つ虚数単位の變種 ε を持つ数である。双対数に対して、通常の実数と同様に、四則演算や指数関数の定義を自然に拡張することができる。双対数への機械的な置き換えによって得られたアルゴリズムが計算する結果の虚部を取り出すことで、求める微分値、す

なわち式 (18) 最右辺の第 1 項を得ることができる。^(注2)

上記 2 つの導出いずれを用いるにせよ、結果として得られるアルゴリズムは本質的に同一のものとなる。結果、得られるアルゴリズムを Algorithm 2 に示す。

最後に、Algorithm 1 の 8 行目は、現在のパラメタ値、目標関数値、パラメタ勾配から次に目標関数とパラメタ勾配を求めるべきパラメタ値を返すものである。この `update-parameters` の具体的な実装は、最適化に用いるアルゴリズム（たとえば最急降下法が共役勾配法か、など）やラインサーチアルゴリズムなどから決定されるものである。

本節で述べた最適化手順で必要となる各計算の時間計算量・空間計算量は比較的妥当なものである。 $\mathbf{f}_{(2, \mathbf{x})}$ および $\frac{\partial \mathbf{f}_{(2, \mathbf{x})}}{\partial \mathbf{h}_1}$ の計算を除いて、時間計算量は同じ素性数からなる CRF の最適化の高々定数倍であり、空間計算量も、各 DAG の各辺において、発火する素性の数だけ一時的な値を保持する必要があるだけである。また、 $\mathbf{f}_{(2, \mathbf{x})}$ および $\frac{\partial \mathbf{f}_{(2, \mathbf{x})}}{\partial \mathbf{h}_1}$ の計算に関しても、実際のタスクにおいては、 $\mathbf{f}_{(2, \mathbf{x})}$ は限られた場所の限られた種類の周辺化出力素性にのみ依存する形で定義されると考えられるため、これらの計算も非常に妥当な計算量で実行できると考えてよいであろう。

以上までは、 L_2 の入力素性が L_1 の出力素性に依存する最も単純な場合についてのみに言及してきた。しかしながら、前述の内容はすべて、より複雑な系列ラベリング間の依存関係に対して容易に一般化できる。すなわち、ある系列ラベリングが複数の系列ラベリングの出力に同時に依存する場合、またはある系列ラベリングの出力に依存した系列ラベリングがさらに別の系列ラベリングに依存される場合にもこれまで述べてきた定式化および最適化アルゴリズムを容易に一般化することができる。最終的には、出力素性と入力素性との間の依存関係が非循環有向グラフをなしているようなあらゆる状況にまで一般化できる。

4. 評価実験

本稿で提案したモデルの有効性をほかの手法と比較するため実験を行った。対象としたタスクは、英文（英単語の列）に対

(注1): 用語の定義などに関する詳細は、Li [6] やその参考文献などを参照されたい。

(注2): 自動微分法を実行する目的のためにこの双対数を用いることができることに関してはたとえば Berz [9] などを参照のこと

Algorithm 2 素性間共分散を求める前向き後ろ向きアルゴリズム

Input: $\mathbf{f}_{\langle 1, \mathbf{x} \rangle}$, $\phi_{e_1} \stackrel{\text{def}}{=} \exp(\boldsymbol{\theta}_1 \cdot \mathbf{f}_{\langle 1, e_1, \mathbf{x} \rangle})$, $h'_{e_1} \stackrel{\text{def}}{=} \frac{\partial \mathcal{L}_2}{\partial \mathbf{h}_{\langle 1, e_1 \rangle}} \cdot \mathbf{h}_{\langle 1, e_1 \rangle}$

Output: $q_{k_1} = \text{Cov}_{P(\mathbf{y}_1 | \mathbf{x})} \left[H'_{\langle 1, \mathbf{y}_1 \rangle}, F_{\langle 1, k_1, \mathbf{y}_1, \mathbf{x} \rangle} \right]$ ($\forall k_1 \in \mathcal{K}_1$)

- 1: for $\forall v_1 \in \text{src}(G_1)$, $\alpha_{v_1} \leftarrow 1$, $\alpha'_{v_1} \leftarrow 1$ ($\text{src}(G_1)$ は G_1 のソース集合を表す)
 - 2: for all $v_1 \in V_1$ in a topological order do
 - 3: $\alpha_{v_1} \leftarrow \sum_{x \in \text{prev}(v_1)} \phi_{(x, v_1)} \alpha_x$, $\alpha'_{v_1} \leftarrow \sum_{x \in \text{prev}(v_1)} \phi_{(x, v_1)} \left(h'_{(x, v_1)} \alpha_x + \alpha'_x \right)$ ($\text{prev}(v_1) \stackrel{\text{def}}{=} \{x \in V_1 \mid (x, v_1) \in E_1\}$)
 - 4: end for
 - 5: $Z_1 \leftarrow \sum_{x \in \text{snk}(G_1)} \alpha_x$ ($\text{snk}(G_1)$ は G_1 のシンク集合を表す)
 - 6: for $\forall v_1 \in \text{snk}(G_1)$, $\beta_{v_1} \leftarrow 1$, $\beta'_{v_1} \leftarrow 1$
 - 7: for all $v_1 \in V_1$ in a reverse topological order do
 - 8: $\beta_{v_1} \leftarrow \sum_{x \in \text{next}(v_1)} \phi_{(v_1, x)} \beta_x$, $\beta'_{v_1} \leftarrow \sum_{x \in \text{next}(v_1)} \phi_{(v_1, x)} \left(h'_{(v_1, x)} \beta_x + \beta'_x \right)$ ($\text{next}(v_1) \stackrel{\text{def}}{=} \{x \in V_1 \mid (v_1, x) \in E_1\}$)
 - 9: end for
 - 10: for $\forall k_1 \in \mathcal{K}_1$, $q_{k_1} \leftarrow 0$
 - 11: for all $(u_1, v_1) \in E_1$ do
 - 12: $p \leftarrow \phi_{(u_1, v_1)} (\alpha_{u_1} \beta'_{v_1} + \alpha'_{u_1} \beta_{v_1}) / Z_1$
 - 13: for $\forall k_1 \in \mathcal{K}_1$, $q_{k_1} \leftarrow q_{k_1} + p f_{\langle 1, k_1, e_1, \mathbf{x} \rangle}$
 - 14: end for
-

して、品詞ラベル列の付与、および基本句切り出しを行うタスクである。

基本句切り出しとは、英単語列の各連続部分列に対して「名詞句」「動詞句」などといった句カテゴリを付与するタスクである。このタスクは、各英単語に対して各基本句の先頭をなすことを示す *Begin-Category*、各基本句の内部にあることを示す *Inside-Category*、どの基本句にも属さないことを示す *Other* というラベルを付与していく系列ラベリングタスクとしてモデル化することができる。[10] *Category* には、具体的には “NP” (noun phrase, 名詞句)、“VP” (verb phrase, 動詞句) などといった句カテゴリが入る。このタスクを行うにあたっては、まず入力単語列に対して品詞ラベル付けがなされるのが一般的である。すなわち、英単語列に対する品詞ラベル付けの解析出力に依存する形で基本句切り出しが行われる。したがって、本稿で対象とする、ある系列ラベリングの出力に依存して別の系列ラベリングを実行するという形態を持つ典型的なタスクとなる。

実験に用いたデータは Penn Treebank プロジェクト [11] によって正解ラベル付けされた Wall Street Journal の英文である。データは 10948 文、259104 単語からなる。これを 8936 文、211727 単語からなる訓練データと、2012 文、47377 単語からなる評価データに分割した。各単語に対して可能な品詞ラベルの総数は 45、基本句切り出しに用いられる可能なラベルの総数は 23 である。

提案したモデルを従来の 2 つの系列ラベリングの手法と比較した。比較対象の 1 つ目は 1-best パイプラインに相当する手法である。正解データから各文と正解品詞ラベル系列の対を取り出して品詞ラベル付けのための CRF を独立に学習する。次に、各入力文と、学習した CRF が各入力文に対して出力する最尤品詞ラベル系列を入力とし、基本句切り出しラベル系列を推定する CRF を独立に学習した。これを “CRF + CRF” とする。比較対象の 2 つ目は、先と同様に品詞ラベル付けのため

の CRF を独立に学習したのち、品詞ラベル付けモデルから得られる周辺化出力素性から基本句切り出しラベル系列を推定する CRF のための入力素性を構築する。ただし、基本句切り出しを行う CRF の学習の際には品詞ラベル付けの CRF のモデルパラメタを固定した。これを “CRF + CRF-MF” とする。このモデルは Bunescu [4] の手法を本タスクに適用することに相当する。最後に、本稿で提案したモデルを本タスクに適用したものに相当する “CRF + CRF-BP” を用意する。“CRF + CRF-BP” は “CRF + CRF-MF” と同じく周辺化出力素性を用いる点で同一であるが、両 CRF のモデルパラメタを 3. 節で述べた方法で同時最適化するものである。“CRF + CRF-BP” においては、同時最適化の際の目標関数が全域で凸であるとは限らないためパラメタの初期値の決定に留意する必要がある。本実験では、“CRF + CRF-MF” (このモデルにおける 2 つの CRF の目標関数はともに全域で凸である) で推定したモデルパラメタを同時最適化の際の初期パラメタとした。各々のモデルにおける超パラメタであるパラメタ事前分布の分散を最大事後確率推定で求めるために、訓練データを 5 分割することによる交差検定を行った。ただし、超パラメタの決定は品詞ラベル付けモデル、基本句切り出しモデル双方の独立した学習で決定した値を用いる。“CRF + CRF-BP” の同時学習の際には、推定した超パラメタは変更しないこととした。モデルの素性として使用した素性テンプレートの一覧を表 1 に示す。本稿では辺に素性が定義される場合のみに言及してきたが、評価実験では頂点上の素性も用いている。

各モデルを本タスクに適用した実験の結果を表 2 に示す。表 2 から、品詞ラベル付けおよび基本句切り出し双方において、本稿で提案したモデルと最適化手法が、単純な 1-best パイプライン (CRF + CRF) および前段階の出力候補集合を確率的な素性の形式で考慮する手法 (CRF + CRF-MF) の双方よりも高精度な結果となり、従来の手法と比較して本手法の有効性

頂点素性	辺素性
頂点がソースかどうか	辺のテールの頂点がソースかどうか 辺のヘッドの頂点がシンクかどうか 辺に対応する隣接品詞ラベル順序対 ‡
頂点がシンクかどうか	
同一時間スライスの入力単語文字列自身	
同一時間スライスの入力単語の接尾文字列 n 文字 ($n \in [1, 2, 3]$)	
入力単語文字列の先頭が大文字 †	
入力単語文字列の全体が大文字 †	
入力単語文字列が品詞 T を持ちうる † ($T \in \{ \text{全ての可能な品詞ラベル} \}$)	
入力単語文字列が数字を含む †	
頂点に対応する品詞ラベル ‡	

表 1 用いた素性テンプレート。頂点素性是对应する頂点のラベル（品詞ラベルまたは基本句切り出しラベル）素性と組み合わせ、辺素性是对应する隣接ラベル順序対の素性と組み合わせる。† の素性は前後 2 単語分の幅の各々の時間スライスにおいて適用する。‡ の素性は品詞ラベル付けには使用せず、“CRF + CRF-MF”、“CRF + CRF-BP” においては出力素性として周辺化する。

	CRF + CRF	CRF + CRF-MF	CRF + CRF-BP
品詞ラベル付与	0.956	(0.956)	0.958
基本句切り出し	0.921	0.927	0.931

表 2 各モデルの精度 (F-値)

が示される結果となった。特に、“CRF + CRF-MF” と比較して、微小ながら品詞ラベル付けに対する精度も向上している。このことから、本稿で提案した同時学習を行う最適化アルゴリズムにより後ろ向きの情報伝搬が達成できているといえる。

5. ま と め

本稿では、系列ラベリングによるラベル系列の推定を階層的に複数接続する状況に対して、前段階の出力に関する素性を周辺化して次段階の解析に用いる手法を採用し、この手法においては、次段階のモデルが、素性の周辺化を通して前段階のモデルパラメタに間接的に依存するとみなす提案を行った。さらに複数のモデルパラメタを同時最適化する手順を提案し、実際のタスクで提案の有効性を確認した。今後は、 k -best 法や同時学習など、他の競合する手法と精度および実行時間の両面で比較し、本手法の有効性をより徹底して検証していきたい。また、本手法に基づいて、準マルコフモデルや CKY アルゴリズムによって導出される木の集合に対する分類問題を階層的に接続する状況に対しても同時学習を実行し、その有効性を検証していきたいと考える。

文 献

- [1] J. Lafferty, A. McCallum and F. Pereira: “Conditional random fields: Probabilistic models for segmenting and labeling sequence data”, Proceedings of the Eighteenth International Conference on Machine Learning, pp. 282–289 (2001).
- [2] B. Taskar, C. Guestrin and D. Koller: “Max-margin Markov networks”, Advances in Neural Information Processing Systems 16 (2003).
- [3] J. Finkel, C. Manning and A. Ng: “Solving the problem of cascading errors: Approximate bayesian inference for linguistic annotation pipelines”, Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing, pp. 618–626 (2006).
- [4] R. Bunescu: “Learning with probabilistic features for im-

proved pipeline models”, Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing, pp. 670–679 (2008).

- [5] C. Sutton, A. McCallum and K. Rohanimanesh: “Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data”, The Journal of Machine Learning Research, **8**, pp. 693–723 (2007).
- [6] Z. Li and J. Eisner: “First-and second-order expectation semirings with applications to minimum-risk training on translation forests”, Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1, pp. 40–51 (2009).
- [7] R. Wengert: “A simple automatic derivative evaluation program”, Communications of the ACM, **7**, 8, p. 464 (1964).
- [8] G. Corliss, C. Faure and A. Griewank: “Automatic differentiation of algorithms: from simulation to optimization”, Springer Verlag (2002).
- [9] M. Berz: “Automatic differentiation as nonarchimedean analysis”, Computer Arithmetic and Enclosure, pp. 439–450 (1992).
- [10] L. Ramshaw and M. Marcus: “Text chunking using transformation-based learning”, Proceedings of the Third ACL Workshop on Very Large Corpora Cambridge MA, USA, pp. 82–94 (1995).
- [11] M. Marcus, M. Marcinkiewicz and B. Santorini: “Building a large annotated corpus of English: The Penn Treebank”, Computational linguistics, **19**, 2, p. 330 (1993).