GibbsBoost による類似文章検索の検討

山田一郎† 中田洋平‡ 松井淳†‡ 松本隆‡ 三浦菊佳†
住吉英樹† 八木伸行†
†ＮＨＫ放送技術研究所, ‡早稲田大学大学院理工学研究科
E-mail: yamada.i-hy@nhk.or.jp

1 はじめに

近年、放送局では番組を蓄積・管理するシステムが普及し、NHK においても NHK アーカイブスに約 59 万本もの放送された番組が蓄積されている。これらのコンテンツを有効活用するためには、番組のどの区間に何が映っているかというメタデータが重要な役割を果たす。これまでに我々は、番組のクローズドキャプションを対象として定型表現を含む文章区間を抽出する手法を提案した[1]。テレビ番組のナレーションでは、「場所紹介」や「人物紹介」など特定の事柄を表現するために同じような言い回しが多用されるため、定型表現を抽出することにより、対応する番組映像区間に映像内容のメタデータを付与することができる。この手法では、複数文からなる文章から構文解析結果の木構造を生成して、木構造間の類似性を評価することにより定型表現か否かを判定する。類似性評価の処理では、木構造から生成した大量の部分木を弱学習器として利用し AdaBoost アルゴリズムにより学習を行う。この際、比較対象となる木構造から生成される部分木の数は膨大な数とり、最終的には類似性評価に使われない無駄な部分木も弱学習器として生成するため効率が悪いという問題が残されていた。

本稿では、効果的かつ効率的に部分木をサンプリングして学習する GibbsBoost アルゴリズム[2][3]を用いて、特定の事柄を表現するための言い回しを含む文章か否かを判別する手法を提案する。構文木から部分木を生成する際、先見的情報により効率的に部分木の選択を行う。学習処理では、複数の弱学習器の系列から逐次モンテカルロ法によりサンプリングして解を導く。

以下、2 章で GibbsBoost アルゴリズムの説明を行い、3 章では GibbsBoost アルゴリズムを類似文章検索に応用する手法について述べる。４章では実際のクローズドキャプションを利用した類似性評価実験を行い、最後にまとめと今後の課題について言及する。

2 GibbsBoost アルゴリズム

Boosting アルゴリズムは教師有り学習手法の一つで、2 値の出力を持つ学習データが与えられたとき、逐次的に学習データの重みを変化させながら誤り率を最小化する弱学習器が選択される。この弱学習器を組み合わせて精度の高い学習器を構成し、観測データが 2 値の何れであるかを判別する。判別関数は式(1)で表される。

$$F(x;\Theta_t)=\sum_{t'=1}^{t}\alpha_{t'}h(x;\theta_{t'}) \qquad (1)$$

ここで、x は観測データ、h は弱学習器、α_t は t 番目の弱学習器の信頼度、θ_t は t 番目にどの弱学習器を選択するかを決めるパラメータである。また、$\Theta_t := (\alpha_1,\ldots,\alpha_t,\theta_1,\ldots,\theta_t)$ とする。学習処理では、入力 x_i と出力 y_i からなる学習データ $\{y_i,x_i\}_{i=1}^{N}$ が与えられた時、損失関数 $\sum_{i=1}^{N}L(y_iF(x;\Theta_t))$ を最小化する Θ_t を $t=1,\ldots,T$ について逐次的に求める。Θ_t の決定により、入力 x に対する判別関数 F の値が 0 より大きい場合+1 を返し、0 以下の場合-1 を返す 2 値判別器を実現できる。

GibbsBoost アルゴリズムも Boosting アルゴリズムの一つであり、弱学習器を効果的かつ効率的にサンプリングすることで弱学習器の系列を複数生成する。GibbsBoost アルゴリズムでは、Boosting アルゴリズムの損失関数に対応するエネルギー関数 $L(z)$を用いて、パラメータ Θ_t に対する確率分布 $P_t(\Theta_t)$ を(2)式により定義する。

$$P_t(\Theta_t)\propto\pi(\Theta_t)\prod_{i=1}^{N}\exp\left(-\beta_t L\left(y_i\frac{F(x_i;\Theta_t)}{\sqrt{t}}\right)\right) \qquad (2)$$

式(2)では、損失関数 $L(z)$からの影響が t に比例して増えないよう $\sqrt{t}$ により抑制している。β_t は、統計力学における温度の逆数を示す係数であり、確率分布 $P_t(\Theta_t)$ の分散を抑制する。β_t が大きい場合、Θ_t は損失関数の和が小さくなるような値に集中し、β_t が小さい場合、$\pi(\Theta_t)$ と似た分布となる。本実験では Cauchy annealing をベースとした値 $\beta_t=0.7(t+1)$ を利用した。$\pi(\Theta_t)$ はパラメータ Θ_t に対する事前確率分布であり、式(3)により定義される。

弱学習器の各系列数(t = 1 to T)において、Step1～Step3 を繰り返す

Step1: t-1 個の弱学習器が線形結合された系列 j (j = 1～M)に対して、t 番目の弱学習器を提案分布 $Q(\Theta_{t-1})$ に基づきサンプリング

$$(\alpha_t^{(j)}, \theta_t^{(j)}) \sim Q(\alpha_t, \theta_t; \Theta_{t-1}^{(j)})$$

Step2: Step1 において選択された系列 $\Theta_t^{(j)}$ の Importance weight $w^{(j)}$ を計算

$$w^{(j)} \propto \frac{P_t(\Theta_t^{(j)}; \beta_t)}{P_{t-1}(\Theta_{t-1}^{(j)}; \beta_{t-1}) Q(\alpha_t^{(j)}, \theta_t^{(j)}; \Theta_{t-1}^{(j)})}$$

ここで、$\sum_{j=1}^{M} w^{(j)} = 1$

Step3: 確率 $w^{(j)}$ によって $\Theta_t^{(j)}$ をリサンプリング

図1. GibbsBoost アルゴリズムにおける サンプリング処理手順

$$\pi(\Theta_t) = \prod_{t'=1}^{t} \pi_\theta(\theta_{t'}) \pi_\alpha(\alpha_{t'}) \quad (3)$$

$\pi_\theta(\theta_{t'})$ は、t' 番目の弱学習器の候補を選択する事前分布であり、先見的情報に基づいて決定する。$\pi_\alpha(\alpha_{t'})$ は t' 番目の弱学習器の信頼度に対する事前分布であり、ここでは正規分布と定める。$\pi(\Theta_t)$ の詳細は 3 章で説明する。

GibbsBoost アルゴリズムにおける判別関数は、使用する弱学習器の系列数を T 個としたとき、$F(x;\Theta_T)$ に対してパラメータ Θ_T に対する確率分布 $P_T(\Theta_T)$ による期待値により、式(4)で定義される。

$$F_{ave,T}(x) = \int F(x;\Theta_T) P_T(\Theta_T) d\Theta_T \quad (4)$$

式(4)は、解析的に解を求めることが困難である。そこで逐次モンテカルロ法を利用し、Θ_T を有限個数だけサンプリングする。サンプリングの処理手順を図 1 に示す。逐次モンテカルロ法では、パラメータ Θ_t のサンプリングと、その重み(importance weight)の計算のために提案分布 $Q(\Theta_t)$ を使用する。$Q(\Theta_t)$ の値が大きい部分から多くサンプリングし、$Q(\Theta_t)$ の値が小さい部分からはあまりサンプリングを行わない。そして、$Q(\Theta_t)$ をもとに取り出した M 個のサンプル $\Theta_t^{(j)}$, j=1～M に対する重み $w^{(j)}$ (importance weight)を計算する。この重み $w^{(j)}$ は、選ばれた弱学習器の系列に対する $P_t(\Theta_t)$ の値により算出される。この重み $w^{(j)}$ の値を利用して、選ばれた $\Theta_t^{(j)}$ からリサンプリングを行う。大きな重みを持つ $\Theta_t^{(j)}$ は何度も選択され、小さな重みを持つ $\Theta_t^{(j)}$ は選択されない。この処理を、線形結合する弱学習器数 t=1～T のそれぞれの場合において逐次的に行うことにより、T 個からなる弱学習器系列が M 個生成される。これらの系列がサンプリングされた結果となる。

弱学習器の選択を決定する提案分布は $Q(\theta_t)$=$\pi_\theta(\theta_t)$ とし、選ばれた弱学習器の信頼度を決定する提案分布 $Q(\alpha_t)$ は学習データに対するエラーレートなどから算出する。

M 個のサンプリング結果を利用して、$F_{avs,T}(x;\Theta_T)$ の和を計算する。M 個のサンプルを $\{\Theta_T^{(j)}\}_{j=1}^{M}$ とすると、式(4)は式(5)で近似される。

$$F_{ave,T}(x) \approx \frac{1}{M} \sum_{j=1}^{M} F(x;\Theta_T^{(j)}) \quad (5)$$

この値の正負を判定基準とすることにより、2 値判別が可能となる。

3 定型的な文章か否かの判別処理

番組のクローズドキャプションでは、特定の事柄を表現するために同じような言い回しが多用される。例えば、表1に示すクローズドキャプションでは、矩形で囲まれた部分が「場所」を映像とともに説明している。最初に体言止めにより「オンフルール」という町の位置情報を説明し、次に町の詳細を断定の助動詞「です」を使って説明している定型的な表現である。本処理では与えられた複数文からなる文章が、同じような言い回しからなる定型表現区間であるか否かを判定する。まず、複数文からなる文章から構文解析結果の木構造を抽出し、次に、木構造間の類似性を評価指標とする弱学習器を生成する。GibbsBoost アルゴリズムにより、生成された弱学習器をサンプリングし、式(5)の関数を生成する。以下に部分木抽出、弱学習器生成、そして、GibbsBoost による学習について記す。

表1 クローズドキャプション例（矩形で囲まれた部分は「場所」を説明する定型的な表現）

提示時間	クローズドキャプション
08:29:09	まっ こんなとこですかね
08:29:12	やっぱり 絵を描かなくてよかったかもしれませんね
08:29:46	セーヌ川を挟み ル･アーブルの対岸に位置する港町 オンフルール
08:29:53	今なお中世の古い家並みが残る 町です
08:29:59	18歳の時 モネは パリに出て画家を 目指しますが 美術学校の 入学試験に合格しませんでした

3．1 部分木抽出

入力テキストを一文ごとに構文解析して、各ノードを文節により構成する構文木を生成する。各文の根ノードの親ノードに最上位ノードを生成し、最上位ノードから各文の構文木へは順序付きのアークで結んだ木構造を生成する。順序付きアークは文の出現順序を考慮した木構造間の類似度評価で利用する。表1の矩形で囲まれた区間の入力テキストを木構造に変換した例を図2に示す。次に、学習データ中の正例として与えられた木構造からキーとなる単語を含む部分木を生成する。今回の実験では「場所」を映像とともに説明する言い回しの有無を判定対象としているため、「オンフルール」のような地名を表す単語は「地名」、「町」のような地名の言い換え表現を表す単語は「地言換」というラベルで抽象化した。また、部分木の作成の際にノードの飛び越えを許し、飛び越えたノードは”+”の記号で置き換え1つ以上のノードとのマッチングを許した。図2に示した木構造から生成される部分木の一部を図3に示す。

3．2 弱学習器生成

抽出した部分木と、学習データに含まれるテキストから生成される木構造との類似度は、部分木に含まれる葉ノードから根ノードまでの全リスト構造を抽出し、その各リスト構造が対象とする木構造に含まれる割合を基準として定義する。部分木rと木構造xの類似度$sim(r,x)$は式(6)とする。

$$sim(r,x)=\frac{1}{N(r)}\sum_{r_i\in r}\frac{1}{K(r_i)}\sum_{sr\in r_i}\max_{sx\in x}(C^d\times sim'(sr,sx)) \tag{6}$$

r_i：部分構造rに含まれるi番目の文
sr：r_iに含まれる葉ノードから根ノードまでのリスト
sx：xに含まれる葉ノードから根ノードまでのリスト
$sim'(sr,sx)$： srがsxに含まれる割合。リストに含まれる主辞と付属語を分割して計算。
$N(r)$：rに含まれる文数
$K(r_i)$：r_iに含まれるリスト数
C：文の出現順序の差に与えるペナルティ値（本実験では0.5とした）
d：文の出現順序の差

文の出現順序の差は、複数文からなる構文木と部分木を比較するときに生じるもので、複数の組み合わせの可能性がある場合は、その最大値を$sim(r,x)$とする。類似度が一定値以上か否かを判断基準とすることにより、入力xに対して部分木rと閾値φ_rを変数に持つ弱学習器$h(x;r,\varphi_r)$を生成する。

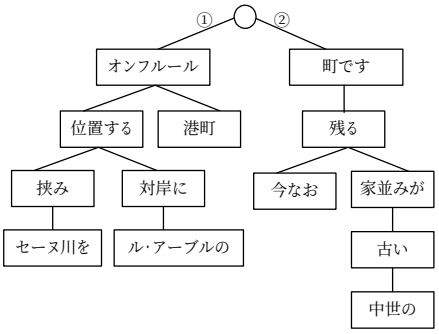

図2. 木構造生成例

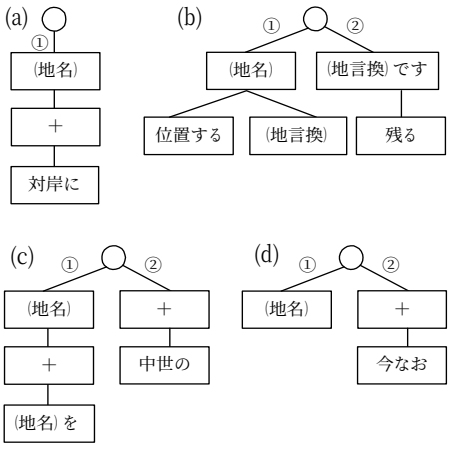

図3. 木構造から抽出された部分木（一部）

3．3 GibbsBoost による学習

部分木生成時にキーとなる地名以外にいくつのノード（文節）を利用するかにより、弱学習器の特徴が決定するが、利用するノード数が多い場合は計算量が膨大になる。もし、100個のノードから50個抽出する場合、その組み合わせ数は$1.0\text{x}10^{29}$個を超え計算が困難と考えられる。そこで、効率的な部分木のサンプリングが必要となる。

図2などに示されるような構文木では、各文の根ノードに主節の述部があり、その下のノードには、主節の述部に直接係る連体節または主節の格要素が位置する。これらは文を比較する上で重要な要素と考えられる。そこで、根ノードに近いノードほど選択される確率が高くなり、さらに、選択されたノード間の距離が近いほど選択される確率が高くなるような事前分布$\pi_\theta(\theta)$を式(7)の通り定義する。

$$\pi_\theta(\theta)\approx\prod_{n\in\theta}C_1^{depth(n)}\times\prod_{n_1,n_2\in\theta,n_1\neq n_2}C_2^{length(n_1,n_2)} \tag{7}$$

$depth(n)$：ノードnの根ノードからの深さ
$length(n_1,n_2)$：ノードn_1とノードn_2間のノード数
C_1：ノードの深さに対するペナルティ(本実験では0.9)

C_2：2 つのノード間の距離に対するペナルティ(本実験では 0.95)

学習データの正例の構文木から生成された大量の部分木（弱学習器）に対して、式(7)の値により一定数 M 個をサンプリングし(図 1 Step1)、次に選択された M 個に対して Importance weight を計算する(図 1 Step2)。この Importance weight の値によって、利用する弱学習器を決定する。Importance weight の値が高い系列ほど、次の処理でも利用される確率が高くなる。さらに、改めて M 個の弱学習器をサンプリングし、同様の処理を T 回繰り返すことにより、T 個の弱学習器の系列が M 個出来る。図 4 に GibbsBoost アルゴリズムの概念図を示す。ここでは、網掛けの弱学習器が最終的に選択された弱学習器となっている。この系列を利用して、最終的に、式(5)により 2 値判別を行う。

4 「場所」を説明する定型表現判別実験

NHK で放送された紀行番組「わが心の旅」のクローズドキャプションを対象として、「場所」を映像とともに説明している定型的な文章と、場所を表す単語があるが場所を映像とともに説明していない文章の判別実験を行った。48 番組に対して人手により「場所」を映像とともに説明している定型的な文章 154 区間を抜き出し学習データの正例とした。負例も、正例と同数だけ人手により無作為に抽出した。この学習データを 2 つに分け、一方を学習データ、他方をテストデータとしたクロスバリデーションによる実験を行った。使用するノード数を 3 個とした時、1 回の試行では、平均 10655 個の弱学習器が生成された。サンプリングする弱学習器の数を M=500、弱学習器の系列の長さを

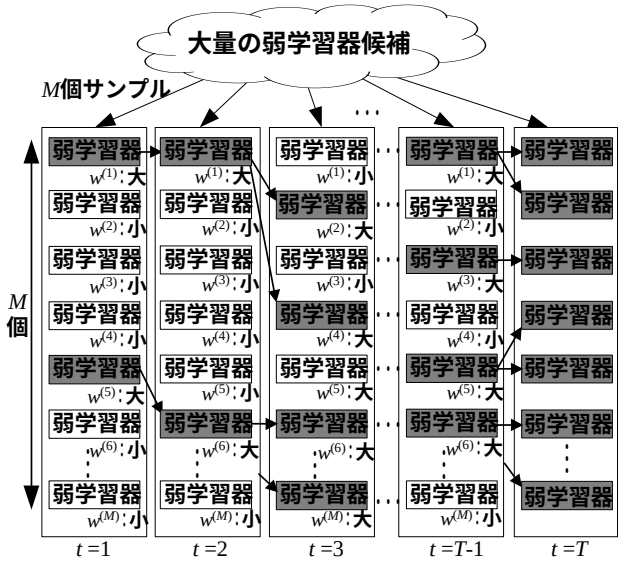

図 4. GibbsBoost アルゴリズムの概念図

T=500 とし、サンプリング時の乱数の影響を吸収するために 5 回の試行を行った。比較対象として、我々がこれまでに提案している AdaBoost を利用した手法による実験を行った[1]。この手法では、弱学習器生成までは同じ処理を行い、重み付きの学習データに対する誤り率を最小とする弱学習器を選択する。この手法でも、弱学習器の系列の長さを T=500 とした実験を行った。分類精度の平均を表 2 に示す。

表 2　場所説明単語判定評価結果

	定型的な文章	定型的でない文章	全体
提案手法 (GibbsBoost)	131.2/154 （85.2%）	142.4/154 （92.5%）	273.6/308 （88.8%）
従来手法 (AdaBoost)	124/154 （80.5%）	149/154 （96.8%）	273/308 （88.6%）

Xeon™ 2.80GHz x 2 の PC を用いた各手法における平均学習時間(CPU Time)は、以下の通りであった。

提案手法(GibbsBoost)　4 分 30 秒
従来手法(AdaBoost)　325 分 48 秒

提案手法は全ての弱学習器を考慮する従来手法と同程度の精度を保ちながら、学習時間は 80 倍以上の速さを実現していることがわかる。

5 まとめ

本稿では、GibbsBoost アルゴリズムを用いて、特定の事柄を表現するための言い回しを含む文章か否かを判別する手法を提案した。実験では、従来の AdaBoost を用いる手法と同程度の精度を保ちながら、学習時間を大幅に短縮できることが確認できた。今後は、実験的に決定したパラメータの最適化による精度改善をはかるとともに、他の定型表現検索へ応用していく予定である。

[参考文献]

[1] 山田, 三浦, 住吉, 八木, 奥村, 徳永: AdaBoost を利用した字幕テキストからの定型表現文章区間抽出, 情処学会研究報告 NL174, Vol.2006, No.82, pp25-30(2006)

[2] Y. Nakada, Y. Mouri, Y. Hongo, T. Matsumoto: Gibbsboost: a Boosting Algorithm using a Sequential Monte Carlo Approach, Proceedings of the 2006 16th IEEE Signal Processing Society Workshop, pp259-264(2006)

[3] 木村, 松井, 中田, 松本: GibbsBoost による正面顔画像検出:事前情報を考慮する Bayes 的アプローチ, 第 5 回情報科学技術フォーラム FIT2006, I-008, pp.17-18(2006)