社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

[特別講演] カーネル法による構造データの解析

鹿島　久嗣

† 日本アイ・ビー・エム株式会社 東京基礎研究所　〒 242–8502 神奈川県大和市下鶴間 1623–14
E-mail: †hkashima@jp.ibm.com

あらまし　配列や木, グラフなどの構造を持ったデータの解析法として, サポートベクターマシンに代表されるような, カーネル法に基づくアプローチを紹介する. 特に, 構造をもったデータに対するカーネル関数設計の枠組みである, 畳み込みカーネルの考え方を紹介し, これに基づくカーネル関数設計法を, グラフ構造を持ったデータに対するカーネル関数の設計を通して解説する. また, 通常の教師あり分類学習のより一般的な問題として, 構造間のマッピングの問題をとりあげ, この問題へのカーネル法によるアプローチを紹介する.

キーワード　カーネル法,　畳み込みカーネル,　周辺化カーネル,　グラフカーネル, 構造マッピング問題, 隠れマルコフパーセプトロン

Kernel Methods for Analyzing Structured Data

Hisashi KASHIMA

† IBM Tokyo Research Laboratory　Shimotsuruma 1623–14, Yamato-shi, Kanagawa, 242–8502 Japan
E-mail: †hkashima@jp.ibm.com

Abstract　We introduce kernel-based approaches for analyzing structured data such as sequences, trees, and graphs. Especially, we introduce the idea of the convolution kernel that is a general framework for designing kernels for structured data, and give some examples of such kernels. Moreover, we introduce the structure mapping problem that is a generalized problem of the supervised classification problem, and kernel-based approaches for the problem.

Key words　Kernel Methods, Convolution Kernels, Marginalized Kernels, Graph Kernels, Structure Mapping, Hidden Markov Perceptron

1. はじめに

コンピューターにデータからの自動的な学習, すなわち, 機械学習を行わせようとしたとき, その過程は大きく分けて 2 つのステップに分けられる. ひとつ目は対象の表現であり, もうひとつは学習アルゴリズムの選択・適用である. 後者のステップは学習対象がすでに実数ベクトルなどの一般的な表現によって与えられていることを仮定しており, 学習アルゴリズムの研究も, この仮定の上で, 効率や予測性能等を議論することが多い. 一方, 前者のステップは通常, 解析対象の性質を吟味した上で適切な特徴を取り出し, 特徴ベクトルによって表現することが必要である. 従って, 特徴を定義するステップは各々の解析対象に大きく依存し, 一般的なアプローチを決めることは非常に困難である. それでもある程度の普遍性をもつ設計指針や, 自動化された方法があることが望まれる.

近年の情報技術の発展によって, 様々なデータが電子的に管理されるようになり, テキストデータや時系列データからマルチメディアデータ, さらには DNA 配列などの生物学的データまで, 機械学習が対象とするデータも大きな広がりを見せている. こういったデータを対象になんらかの解析を行おうとすると, やはり, 先に述べた特徴ベクトルによる表現は自明ではない. しかしながら, ベクトル形式ではなく, 配列や木, グラフなどのより柔軟で一般的なデータ形式による表現は割と簡単にできることがある. 従って, これらのデータ形式を一般的に扱えるようにするように学習アルゴリズムの方から歩み寄ることは, より簡単にデータ解析を行えるようにする上でとても重要であるといえる.

本稿では, 情報科学において, ベクトルより柔軟で一般的なデータ形式としてよく用いられる配列や木, グラフなどのデータをうまく扱うための機械学習手法, 特に, 近年この分野で注目され 非常に精力的に研究が行われているカーネル法 [1] に基づくアプローチを紹介する.

2. カーネル法とは

はじめに, 2 値分類の学習問題を例として, カーネル法の概念を紹介する.

解析の対象をすべて含んだ集合を X とし, $Y = \{+1, -1\}$ を対象が属するクラスの集合とする. 2 値分類の学習問題は, N 組の対象とクラスの組 $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(N)}, y^{(N)})$(ただし $x^{(i)} \in X$, $y^{(i)} \in Y$) が訓練例として与えられたときに, X から Y への写像 $h : X \to Y$ を例から学習する問題である.

この問題に対するアプローチとして, 通常, 対象 x を D 次元の特徴空間における 1 点として, $\Phi(x) = (\phi_1(x), \phi_2(x), \ldots, \phi_D(x))$ のように, ベクトル形式で表現する. これを x の特徴ベクトル表現と呼ぶ. すると, 2 値分類学習問題は, この特徴空間中に散らばる正例 (クラス"+1"に属する訓練例) と負例 (クラス"−1"に属する訓練例) をうまく分類するような分類境界面を求めることに帰着される. 分類境界面としてよく用いられるのは超平面で, 分類したい対象の特徴ベクトル表現が, 超平面のどちら側にあるかによって, クラスを分類する. 形式的には, 重みベクトル $\mathbf{w} \in \Re^D$ および閾値 $b \in \Re$ を使って,

$$h(x) = \mathrm{sign}\left(\left\langle \mathbf{w}, \Phi(x) \right\rangle + b\right) \tag{1}$$

のような形で表される. ここで, $\langle \cdot, \cdot \rangle$ は内積, sign は引数の符号を返す関数であるとする.

与えられた事例から h を (すなわち $\mathbf{w}$ と b を) 学習するためのアルゴリズムとしては, パーセプトロンがよく知られている. パーセプトロンは, 初期値として $\mathbf{w} = \mathbf{0}$ および $b = 0$ から学習をスタートし, 以下のルールによって訓練例をひとつづつ処理しながら学習を進めていくオンライン型の学習アルゴリズムである. i 番目のデータ $x^{(i)}$ に対し, パーセプトロンは現在の h に基づき, そのデータのクラス予測 $\hat{y}^{(i)} = h(x^{(i)})$ を行う. これが正しいクラス $y^{(i)}$ と異なったときだけ, 以下の規則にしたがって重みベクトルと閾値を更新する.

$$\begin{aligned} \mathbf{w} &\leftarrow \mathbf{w} + y^{(i)}\Phi(x^{(i)}) \\ b &\leftarrow b + y^{(i)}R^2 \end{aligned} \tag{2}$$

ここで R は原点を中心として, データをすべて包含するような最小の球の半径である. 訓練データを完全に分類できる h が存在するとき, パーセプトロンは必ず収束することが知られている.

次に, パーセプトロンをカーネル法の視点から眺めてみる. 重みの更新式 (2) で $y^{(i)} \in \{+1, -1\}$ であることに注意すると, 更新の度に重みベクトルに特徴ベクトル $\Phi(x^{(i)})$ が加えられているか, 引かれているかのどちらかであることがわかる. 従って, 重みベクトルは特徴ベクトルの線形和によって

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i \Phi(x^{(i)}) \tag{3}$$

のように表現できそうである. ここで, α_i は i 番目の訓練例の重みである. これを (1) の式に代入してみると, 次の式が得られる.

$$h(x) = \mathrm{sign}\left(\sum_{i=1}^{N} \alpha_i \left\langle \Phi(x^{(i)}), \Phi(x) \right\rangle + b\right) \tag{4}$$

また更新式 (2) も, すべての i に対し $\alpha_i = 0$ から学習をスタートし,

$$\alpha_i \leftarrow \alpha_i + y^{(i)}$$

のように書きかえることができる.

さて, このように書き変えてみると, 予測や学習において, データの特徴ベクトルが単独で現れることはなく, 常に 2 つの特徴ベクトルの内積の形で現れているところに気づく. この内積を

$$K(x, x') = \left\langle \Phi(x), \Phi(x') \right\rangle$$

と置き換えてみると (4) は,

$$h(x) = \mathrm{sign}\left(\sum_{i=1}^{N} \alpha_i K(x^{(i)}, x) + b\right)$$

と書き換えられる. これは単なる置き換えに過ぎないが, この置き換えによって, 全てのデータアクセスが, この K を通して行われることになる. この K は**カーネル関数**と呼ばれ, このような学習アルゴリズムを総称して**カーネル法**と呼ぶ.

特に, Cortes と Vapnik によって提案されたサポートベクターマシン [2] は, 代表的なカーネル法として知られている. サポートベクターマシンの学習は, 以下の 2 次計画問題として定式化される.

$$\begin{aligned} \text{maximize} \quad & \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j K(x^{(i)}, x^{(j)}) \\ \text{s.t.} \quad & \sum_{i=1}^{N} y_i \alpha_i = 0 \\ & \alpha_i \geqq 0 \end{aligned}$$

一般的なカーネル法において, 最適な $\mathbf{w}$ が (3) のように特徴ベクトルの線形和で表現されることは, Representer 定理と呼ばれる定理によって保証される.

カーネル法は多値分類問題や回帰などの教師あり学習問題に限らず, クラスタリングや主成分分析などの教師なし学習問題など様々な学習問題に適用されている. 様々なカーネル法のバリエーションについては [1] などを参照されたい.

カーネル関数は, 2 つの対象のある種の類似度を定義していると考えることができるが, データアクセス部分をカーネル関数に置き換えることの利点の 1 つとして, 適当な関数 K をカーネル関数として使えるということが挙げられる. そのためには, K が暗に特徴ベクトルの内積になっていることを保証する必要があるが, これは K が半正定, すなわち K が対称 $K(x, x') = K(x', x)$ かつ,

$$\sum_{x \in X} \sum_{x' \in X} w(x) w(x') K(x, x') \geqq 0$$

が任意の $\sum_{x \in X} w(x) < \infty$ である w について満たすことが示されればよい. この条件を満たすカーネル関数は Mercer カーネルと呼ばれる.

さて, 本稿では x や x' が, DNA 配列やテキストなどのように文字列で表される場合, XML や HTML で記述された文書や構文解析木のように木で表される場合, あるいは化合物の分子構造や 3 次元空間内の点集合のようにグラフで表現される場合などのカーネル関数 $K(x, x')$ の設計指針を紹介する.

ところで, 「データにおける構造」といった場合には, 上で挙げたような, 個々の対象そのものに含まれる「内的な」構造と, 対象の間の構造である「外的な」構造がある. 後者は, 例えば Web ページを解析対象としているとき, これらの間のハイパーリンクによって作られるリンク構造であったり, タンパク質を解析対象とする場合にはこれらの間の相互作用の有無であったりする. 「外的な」構造を扱うためのカーネル関数の設計指針としては, 拡散カーネル (Diffusion Kernel) [3] などがあるが本稿では割愛する. 興味のある方は [4] の解説などを参照していただきたい.

3. 構造データを扱うためのカーネル法

この節では, 内部構造をもったデータに対するカーネル関数設計の一般的な枠組みである畳み込みカーネル (Convolution Kernel) [5] と, 配列や木, グラフなどの内部構造をもったデータに対する具体的なカーネル設計の手順を紹介する.

Haussler [5] は, 構造をもったデータの特徴は, その構造に含まれる部分構造が担っていると考え, 構造データ同士のカーネル関数を, 部分構造同士のカーネル関数によって再帰的に定義するというアイデアを提案した (図 1).

例えば, 配列データを生成するモデルとして n 次のマルコフモデルが良く使われるが, これはすなわち, 配列データの性質が連続する長さ n の部分配列によって捉えられるということを仮定していることになる. そこで, 2 つの配列を, それぞれの配列に含まれる全ての長さ n の部分配列の集合として表し, 配列同士のカーネル関数は, これらの部分配列間のカーネル関数の値の和として定義すればよさそうである.

形式的には畳み込みカーネルは

$$K(x, x') = \sum_{s \in S(x)} \sum_{s' \in S(x')} K_S(s, s')$$

のように定義される. ここで $S(x)$ は, x に含まれる部分構造の集合を表し, K_S は 2 つの部分構造の間に定義されるカーネル関数であるとする. この部分構造間のカーネル関数 K_S は, さらに細かい部分構造を用いた畳み込みカーネルとして再帰的に定義される.

より一般的には, x の部分構造 s に対する重み $f(s|x)$ を用いて,

$$K(x, x') = \sum_{s \in S(x)} \sum_{s' \in S(x')} f(s|x) f(s'|x') K_S(s, s') \qquad (5)$$

のようにも書くことができる. 特に,

$$\sum_{s \in S(x)} f(s|x) = 1$$

であるとき, これを**周辺化カーネル** [6], [7] と呼ぶ.

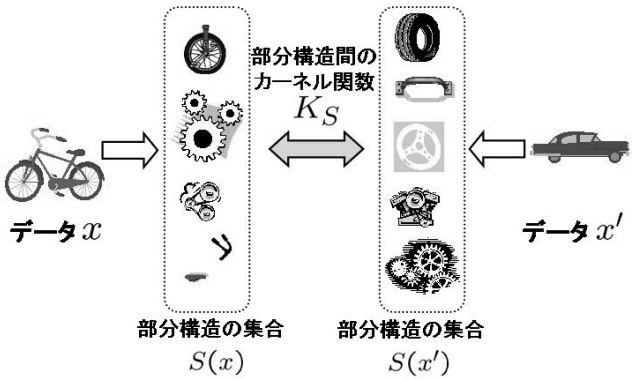

図 1 畳み込みカーネルのイメージ
例えば, 自転車と自動車の間のカーネルを求めるとする. 自転車も自動車もそれぞれ, ギアやタイヤ, エンジンなどのに部品に分解される. カーネル関数は, 自転車のタイヤと自動車のタイヤの間のカーネルや, 自転車のサドルと自動車のドアの間のカーネルなどの, 部品間のカーネルの和として定義される. 部品間のカーネルはさらに小さなネジなどの部品間のカーネルの和として, 再帰的に定義される.

畳み込みカーネルや周辺化カーネルを具体的に適用するためには, 扱いたいデータに応じて部分構造 $S(x)$ と, 部分構造の重み f, カーネル関数を効率よく計算するためのアルゴリズムの設計が必要である. これまでに配列 ([6], [8]〜[10]) や木 ([11], [12]), グラフ ([7], [13]〜[15]) など様々な構造を持つデータに対するカーネル関数とアルゴリズムが提案されており, [1], [16] などに詳細なサーベイがある.

ここでは一例として, グラフ同士のカーネル [7], [13] を紹介する. $x = (V, E)$ と $x' = (V', E')$ は図 2 に示すような, 頂点にラベルの振られた有向グラフであるとする. 各頂点 $v \in V$ にはアルファベット Σ のうちのひとつがラベルとして振られているとし, そのラベルを $\sigma(v)$ と表す. 部分構造の集合 $S(x)$ としては x に含まれる全ての部分グラフが使えそうであるが, 残念ながら全ての部分グラフを $S(x)$ として使うとカーネル関数の計算が NP 困難になってしまう [13]. そこで $S(x)$ として, グラフに含まれるパスを $S(x)$ として用いることにする.

ある長さ n のパス $s = (v_1, v_2, \ldots, v_n) \in S(x), v_i \in V$ に対する重みを, グラフ上のランダムウォークによってパス s が生成される確率

$$f(s|x) = p_s(v_1) p_t(v_2|v_1) p_t(v_3|v_2) \cdots p_t(v_n|v_{n-1}) p_e(v_n)$$

と定義する. ここで, $p_s(v_1)$ は, v_1 からランダムウォークをスタートする確率, $p_t(v_i|v_{i-1})$ は v_{i-1} から v_i に遷移する確率, $p_e(v_n)$ は v_n でランダムウォークが終了する確率とする.

部分構造間のカーネルは以下のように定義される. 2 つのパス $s = (v_1, v_2, \ldots, v_n) \in S(x)$ と $s' = (v'_1, v'_2, \ldots, v'_n) \in S(x')$ に対し,

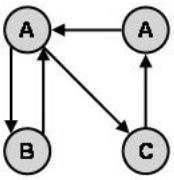

図 2 グ ラ フ

$$K_S(s,s') = \begin{cases} \prod_{i=1}^{n} K_\Sigma(\sigma(s_i),\sigma(s'_i)) & (n = n' \text{のとき}) \\ 0 & (n \neq n' \text{のとき}) \end{cases} \tag{6}$$

のようにさらにラベルごとのカーネル関数 K_Σ の積で定義されるとする．ラベルごとのカーネル関数は, 例えば $\sigma, \sigma' \in \Sigma$ に対し,

$$K_\Sigma(\sigma,\sigma') = \begin{cases} 1 & (\sigma = \sigma' \text{のとき}) \\ 0 & (\sigma \neq \sigma' \text{のとき}) \end{cases} \tag{7}$$

のように定義できる．

さて, グラフに有向サイクルが存在する場合には, ランダムウォークによって生成されうるパスの候補は無限個存在するため (5) の計算を明示的に行うことは不可能である．大抵の畳み込みカーネルでは, 通常, カーネル関数の分解と再帰的表現によって効率的に計算を行う．まず (5) は

$$K(x,x') = p_s(v)p_s(v') \sum_{v \in V} \sum_{v' \in V'} K_V(v,v') \tag{8}$$

$$K_V(v,v') = \sum_{s \in S_v(x)} \sum_{s' \in S_{v'}(x')} \frac{f(s|x)}{p_s(v)} \frac{f(s'|x')}{p_s(v')} K_S(s,s') \tag{9}$$

のように分解できる．ここで $S_v(x)$ は v からスタートするランダムウォークのみによって生成されうるパスの集合とする．(9) は次のように再帰的に書くことができる．

$$K_V(v,v') = p_e(v)p_e(v') + \sum_{\tilde{v} \in V} \sum_{\tilde{v}' \in V'} p_t(\tilde{v}|v)p_t(\tilde{v}'|v')K_V(\tilde{v},\tilde{v}') \tag{10}$$

これは連立方程式となるので, 逆行列の計算や反復計算によって解を求めることができ, これらを (8) に代入することによってカーネル関数の値が求まる．

配列や木は有向グラフの特殊な場合であるので, これらに対するカーネル関数についても同様のアプローチによって設計を行うことができる．特に $S(x)$ がパスの集合である場合のカーネル関数は, 上記と全く同じ議論によって扱うことができる．この場合, 配列や木は有向サイクルを持たないため (10) の再帰式を解く際には, 連立方程式を解く必要はなく, 動的計画法によって $O(|V||V'|)$ で計算をすることができる(注1)．

また, 対象がラベル付き順序木であるような場合には, x に含まれる全ての部分グラフ (木構造をもつ) を $S(x)$ として用いることができる [11], [12](注2)．

(注1)：特殊な場合では, 接尾辞木のような高速なデータ構造を用いて, $O(|V|+|V'|)$ で計算できる ([9], [10], [17])．

(注2)：一般の木の場合には, やはり NP 困難になってしまう．

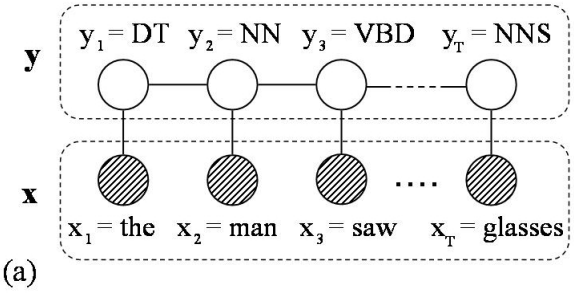

図 3 配列間のマッピング問題のグラフィカルモデル表現．黒い頂点は x, 白い頂点は y を表す．文 “the, man, saw ,$\cdots$, glasses.”は, 単語列 x, 品詞タグ列 “DT, NN, VBD, ..., NNS”は, 各単語に対する品詞の列 y を表す．

4. 構造間のマッピング問題

ここまでは, x のみが構造を持ち, y は 2 値ないし多値や実数値であるような問題を想定してきた．しかしながら, 構造を扱う学習問題を一般化して考えると, x のみならず y の方も構造をもつような問題が考えられる．このような問題を構造マッピングの問題と呼ぶことにする．

実際, このような問題は特に自然言語処理の世界においては昔からよく扱われている問題である．例えば, 品詞付けの問題は, 入力が単語列であり, 出力はそれに対応する品詞の列である．つまり x も y も配列で表されるような問題であり, 固有表現抽出の問題や, タンパク質の 2 次構造予測問題などもこのような問題として捉えられる．また, 構文解析の問題では, 入力が単語列であるのに対し, 出力は構文解析木, すなわち x が配列で y は木となる．

構造マッピング問題に対するアプローチは, 大きく 2 種類に分けられる．ひとつは, x の特徴空間から y の特徴空間へのマッピングを直接学習する方法 [18], [19] であり, もうひとつは x と y の両方をあわせた構造に対する特徴空間を考え, x と y の組み合わせに対して良し悪しの評価を行うような手法 [20]〜[25] である．本稿では, 後者に属する手法の中でも特にシンプルなアルゴリズムである Collins による隠れマルコフパーセプトロン [23] を紹介する．

ここでは簡単な場合として, x と y が同じ長さ T をもつ(注3)配列であるような場合, すなわち $x = (x_1, x_2, \ldots, x_T), x_t \in \Sigma_x$ から $y = (y_1, y_2, \ldots, y_T), y_t \in \Sigma_y$ へのマッピングを求める問題を考える．例えば, 品詞付けなどのタスクにおいて, x_t は t 番目の単語を表わし, y_t は t 番目の単語の品詞を表す (図 3)．

隠れマルコフパーセプトロンも, 分類問題のときと同じく対象を特徴空間におけるベクトルとして表現するが, x だけでなく, x と y をあわせた構造に対する特徴ベクトル $\Phi(x,y)$ として表す点で異なっている．

隠れマルコフパーセプトロンは, パーセプトロンと同じくオンライン型の構造マッピング学習アルゴリズムであり, 訓練例

(注3)：ここでは簡単のため, すべての配列が同じ長さ T を持つかのように記述するが, 実際には同じである必要はない．

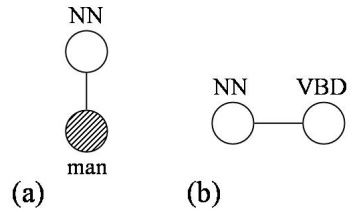

図 4　隠れマルコフパーセプトロンで用いられる部分構造

を 1 つ 1 つ処理することで学習を進める．$x^{(i)}$ が入力として与えられると，隠れマルコフパーセプトロンは現在の重みベクトルに基づき，次式によって予測 $\hat{y}^{(i)}$ を出力する．

$$\hat{y}^{(i)} = \operatorname*{argmax}_{y \in {\Sigma_y}^T} \left\langle \mathbf{w}, \Phi(x^{(i)}, y) \right\rangle \tag{11}$$

重みベクトル $\mathbf{w}$ は，訓練例に対する予測が誤り，すなわち $\hat{y}^{(i)} \neq y^{(i)}$ であったときに，

$$\mathbf{w} \leftarrow \mathbf{w} + \Phi(x^{(i)}, y^{(i)}) - \Phi(x^{(i)}, \hat{y}^{(i)})$$

によって更新される．これを繰り返すことによって，正しいマッピング $(x^{(i)}, y^{(i)})$ が，全ての誤ったマッピング $(x^{(i)}, y), \forall y \neq y^{(i)}$ に対して，

$$\left\langle \mathbf{w}^*, \Phi(x^{(i)}, y^{(i)}) \right\rangle > \left\langle \mathbf{w}^*, \Phi(x^{(i)}, y) \right\rangle$$

となる最適な重みベクトル $\mathbf{w}^*$ に (もしあれば) 収束することが保証される．

さらに (11) はカーネル関数を用いたパーセプトロンの表現を導いたときと同様にして，

$$\begin{aligned} \hat{y}^{(i)} &= \operatorname*{argmax}_{y \in {\Sigma_y}^T} \sum_{j=1}^{N} \sum_{\tilde{y} \in {\Sigma_y}^T} \alpha_j(\tilde{y}) \left\langle \Phi(x^{(j)}, \tilde{y}), \Phi(x^{(i)}, y) \right\rangle \\ &= \operatorname*{argmax}_{y \in {\Sigma_y}^T} \sum_{j=1}^{N} \sum_{\tilde{y} \in {\Sigma_y}^T} \alpha_j(\tilde{y}) K((x^{(j)}, \tilde{y}), (x^{(i)}, y)) \end{aligned} \tag{12}$$

と書き直せる．同様に，重みの更新則も以下のように書き直せる．

- $\alpha_i(y^{(i)}) \leftarrow \alpha_i(y^{(i)}) + 1$
- $\alpha_i(\hat{y}^{(i)}) \leftarrow \alpha_i(\hat{y}^{(i)}) - 1$

ところで (12) において，$\alpha_j(\tilde{y})$ は Σ_y^T 個ある全ての $\tilde{y}$ について定義されているため計算量的に問題がありそうだが，実際には間違えたマッピングについてのみ値がセットされるため，それらだけを考慮すればよい．

さて，(12) はカーネル関数を使って記述されているので，前節で紹介した畳み込みカーネルを K として使うことができそうである．しかし，残念ながら (12) における argmax 操作の計算量は，部分構造の含む y の変数の数に指数的に依存してしまうため，任意の形の部分構造を使った畳み込みカーネルを使うことはできない．そのため，通常は部分構造を小さいサイズに限定し，x_t と y_t のペア (図 4(a)) や，y_t と y_{t+1} のペア (図 4(b)) などの部分構造が使われる．

この問題点に対して，一旦，小さな部分構造のみを用いるマッピングを学習しておいて，それをベースに大きな部分構造を用いるマッピングを学習するという，学習を 2 段階に分けて行うことで解決を図るというアプローチ [4], [22] も提案されている．

5. おわりに

本稿では，特に畳み込みカーネル [5] を中心に構造カーネル法の基本を紹介したが，最後に，構造カーネル法の今後の展望を述べる．カーネル法によって構造データを扱うという考え方は，通常，指数的に大きくなってしまう部分構造の候補数をカーネル関数という形で暗に評価することによって，隠蔽することができるため，構造データに対する美しく，統一されたアプローチを与えるという点で，非常に魅力的な手法であるといえる．

しかしながら，一方で，これらの方法が実用的な面で，本当に使えるか，他の手法に比べて本当に良いのかと考えると，まだまだ，手放しで誉め讃えるというわけにはいかないであろう．カーネル法の最大の特徴である，非明示的に与えられる特徴空間は，任意に高い次元でも多項式時間の学習の礎となっている反面，結果の分類器を人間が見て，知見を得にくいという点でかえってマイナスである．特に，これは構造カーネル法をビジネス用途に用いようとする際には，大きなマイナス要素となるであろう．

このマイナスをカバーするためには「ものすごく速い」か「ものすごく予測性能がいい」かのどちらかは必要であろうが，構造カーネル法はそのどちらでも必ずしも一番だとはいえないのが現状である．

カーネル法はどちらかといえば「遅い」部類に入る学習器であり，現在もその高速化は重要なテーマとして研究が行われている．特に，カーネル予測器は通常 1 回の予測に $O(N)$ の時間を要するため，予測時の高速化は実用面では非常に重要であろう．しかも，構造カーネル法の場合には 1 回のカーネル計算にある程度の計算が必要なため，これは深刻な問題となってくる．興味深い試みとしては，工藤ら [26] の，学習済みのカーネル予測器からパタン発見手法によって重要な部分構造を取り出して，特徴ベクトルを明示的に扱う通常の形の予測器に変換するというアプローチが挙げられる．また，このようなアプローチが使えず，カーネル関数をそのまま使うような場合でも，うまくカーネル関数を変換，管理することで計算量を落とすというアプローチ [27] が研究され始めており，このようなアプローチを構造データに対しても展開できることが期待される．

予測性能について言えば，カーネル法の代表的存在であるサポートベクターマシンでは，マージンと呼ばれる量を最大化することで特徴空間の次元に依存しない良好な予測性能をもつことが理論的にも実験的にも示されてはいるが，やはり，適切な特徴選択あるいは重み付けが予測性能を大きく左右するというのが本当のところであろう．例えば，工藤らのブースティングとパタン発見手法に基づく構造データの分類手法 [28], [29] などは，同じくマージンの最大化に加え，明示的な特徴選択が組み込まれており，また，本質的には NP 困難でありながらも，実用的には多項式時間で動く構造カーネルより高速かつ高性能を誇っている．一方，構造カーネル法における特徴選択の例としては，鈴木ら [30] の試みが挙げられる．彼らは，配列カーネルにおい

て用いる部分構造を χ^2 値によって選択することで, 分類に寄与する特徴のみを用いてカーネルの計算を行うことを提案している. 部分構造の選択は明示的に行われるが, χ^2 値の上界を見積もることで必要な特徴を効率的を効率的に数え上げることを行っている別のアプローチとしては, あくまで特徴選択・重み付けは明示的には行われず, カーネルに含まれるパラメータのチューニングという形で行うということも考えられるであろう. 例えば, 畳み込みカーネルにおける部分構造の重みを学習することによって, 重要な部分構造の重みづけを調整するような方法も有望かもしれない.

文　献

[1] J. Shawe-Taylor and N. Cristianini, Kernel Methods for Pattern Analysis, Cambridge University Press, 2004.

[2] V. Vapnik, Statistical Learning Theory, Wiley, New York, 1998.

[3] R.I. Kondor and J. Lafferty, "Diffusion kernels on graphs and other discrete input spaces," Proceedings of the 19th International Conference on Machine Learning, pp.315–322, 2002.

[4] 鹿島, "カーネル法による構造データマイニング," 情報処理, vol.46, no.1, 2005.

[5] D. Haussler, "Convolution kernels on discrete structures," Tech. Rep. UCSC-CRL-99-10, University of California in Santa Cruz, 1999.

[6] K. Tsuda, T. Kin, and K. Asai, "Marginalized kernels for biological sequences," Bioinformatics, vol.18, no.Suppl. 1, pp.S268–S275, 2002.

[7] H. Kashima, K. Tsuda, and A. Inokuchi, "Marginalized kernels between labeled graphs," Proceedings of the 20th International Conference on Machine Learning, San Francisco, CA, Morgan Kaufmann, 2003.

[8] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins, "Text classification using string kernels," Journal of Machine Learning Research, vol.2, pp.419–444, 2002.

[9] C. Leslie, E. Eskin, J. Weston, and W. Noble, "Mismatch string kernels for svm protein classification," Advances in Neural Information Processing Systems 15, ed. S. Becker, S. Thrun, and K. Obermayer, Cambridge, MA, MIT Press, 2003.

[10] C. Leslie, E. Eskin, and W.S. Noble, "The spectrum kernel: A string kernel for SVM protein classification," Proceedings of the Pacific Symposium on Biocomputing, ed. R.B. Altman, A.K. Dunker, L. Hunter, K. Lauerdale, and T.E. Klein, pp.566–575, World Scientific, 2002.

[11] M. Collins and N. Duffy, "Convolution kernels for natural language," Advances in Neural Information Processing Systems 14, Cambridge, MA, MIT Press, 2002.

[12] H. Kashima and T. Koyanagi, "Kernels for semi-structured date," Proceedings of the 19th International Conference on Machine Learning, San Francisco, CA, pp.291–298, Morgan Kaufmann, 2002.

[13] T. Gärtner, P. Flach, and S. Wrobel, "On graph kernels: Hardness results and efficient alternatives," Proceedings of the 16th Annual Conference on Computational Learning Theory, 2003.

[14] C. Cortes, P. Haffner, and M. Mohri, "Rational kernels," Advances in Neural Information Processing Systems 15, ed. S. Becker, S. Thrun, and K. Obermayer, Cambridge, MA, MIT Press, 2003.

[15] J. Suzuki, T. Hirao, Y. Sasaki, and E. Maeda, "Hierarchical directed acyclic graph kernel: Methods for structured natural language data," Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics, 2003.

[16] T. Gärtner, "A survey of kernels for structured data," SIGKDD Explorations, vol.5, no.1, pp.S268–S275, 2003.

[17] S. Vishwanathan and A. Smola, "Fast kernels for string and tree matching," Advances in Neural Information Processing Systems 15, ed. S. Becker, S. Thrun, and K. Obermayer, Cambridge, MA, MIT Press, 2003.

[18] J. Weston, O. Chapelle, A. Elisseeff, B. Schölkopf, and V. Vapnik, "Kernel depndency estimation," Advances in Neural Information Processing Systems 15, ed. S. Becker, S. Thrun, and K. Obermayer, Cambridge, MA, MIT Press, 2003.

[19] 賀沢, 鈴木, 前田, "マージン最大化に基づく写像近似法SVMAP," 第 6 回情報論的学習理論ワークショップ (IBIS2003), 2003.

[20] A. McCallum, D.Freitag, and F. Pereira, "Maximum entropy Markov models for information extraction and segmentation," Proceedings of the 17th International Conference on Machine Learning, San Francisco, CA, pp.591–598, Morgan Kaufmann, 2000.

[21] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," Proceedings of the 18th International Conference on Machine Learning, San Francisco, CA, pp.282–289, Morgan Kaufmann, 2001.

[22] M. Collins, "Discriminative reranking for natural language parsing," Proceedings of the 17th International Conference on Machine Learning, San Francisco, CA, pp.175–182, Morgan Kaufmann, 2000.

[23] M. Collins, "Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms," Proceedings of the Conference on Empirical Methods in Natural Language Processing, 2002.

[24] Y. Altun, I. Tsochantaridis, and T. Hofmann, "Hidden Markov support vector machines," Proceedings of the 20th International Conference on Machine Learning, 2003.

[25] I. Tsochantaridis, T. Hofmann, T. Joachims, , and Y. Altun, "Support vector machine learning for interdependent and structured output spaces," Proceedings of the 21st International Conference on Machine Learning, pp.823–830, 2004.

[26] T. Kudo and Y. Matsumoto, "Fast methods for kernel-based text analysis," Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics, 2003.

[27] C. Yang, R. Duraiswami, and L. Davis, "Efficient kernel machines using the improved fast gauss transform," Advances in Neural Information Processing Systems 17, 2005.

[28] T. Kudo and Y. Matsumoto, "A boosting algorithm for classification of semi-structured text," Proceedings of the Conference on Empirical Methods in Natural Language Processing, 2004.

[29] T. Kudo, E. Maeda, and Y. Matsumoto, "An application of boosting to graph classification," Advances in Neural Information Processing Systems 17, 2005.

[30] J. Suzuki, H. Isozaki, and E. Maeda, "Convolution kernels with feature selection for natural language processing tasks," Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, 2004.