情報処理学会論文誌　データベース　Vol. 1　No. 3　49–59 (Dec. 2008)

大規模データストリームのための履歴情報を用いたカーネル法の拡張

都築　学[†1]　小西　修[†2]

カーネル法を適用した非線形サポートベクタマシン（SVM）は，最も識別精度に優れたモデルの 1 つである．しかし，時間とともに複雑な変化をともなう実世界のデータストリームに対する識別精度は十分ではなく，これを向上させることは重要な課題である．一方，カーネル法は，適切にカーネル関数を設計することで配列やグラフ等の構造を持つデータを入力に扱えるように拡張され，従来のベクトル形式のデータを入力にするよりも高い識別精度が得られている．そこで本論文では，カーネル法を拡張して，データストリームを構造データと見なして識別する手法を提案する．この構造データは，データストリームのオンラインで到着するデータと，過去いくつかの履歴情報をまとめたものである．我々はこの履歴情報から得られるデータストリームの変化を，識別の特徴に用いるカーネル関数を構築し，これをストリームカーネルと呼ぶ．ストリームカーネルを適用した非線形 SVM は，約 70 万件の実際のクレジットカードデータを用いた不正利用と正常利用の識別実験において，構造データを用いない従来手法よりも誤識別の数を約 40%減少させ，不正利用らしさが上位のデータ中では最大で約 2 倍の不正利用顧客数を検出した．

Adaptive Kernel Methods in Large-scale Data Stream with Historical Information

Manabu Tsuzuki[†1] and Osamu Konishi[†2]

Kernel-based Support Vector Machines (SVMs) have strong theoretical foundations and excellent empirical successes. However, the performance for the real data stream with the evolutionary changes is not enough, thus it is an important challenge to improve the performance. On the other hand, the kernel methods have be extended to input structured data such as sequences or graphs. The classification using the kernel methods with structured form outperform vector form. In this paper, we propose non-linear classification approach to consider data stream as structured data by extended kernel methods. This structured data is constructed with the arrived online data and some historical information of the data stream. Then, the features of the data stream's evolutionary changes are obtained, and kernel function is built with the features. We call this a stream kernel. In the experiment, we apply the stream kernel to non-linear SVM, and differentiate normal use and illegal use for about 700,000 real credit card data. The our approach achieve that a number of the false classification decreased to about 40%, and the illegal users are detected about double at the peak in data seem to be the illegal use.

1. はじめに

近年，金融や流通分野の取引記録や，ネットワーク監視システムの通信記録等のデータストリームが新しいタイプの大規模データとして注目を集めている．データストリームには，日々刻々と生成され，時間とともに複雑な変化をともなう性質があり，この蓄積されたデータをいかに分析し活用するかが重要な課題となっている．そこで，データマイニングや機械学習がその有力な手法となる．

データマイニングや機械学習は，大量のデータから知識やルールを発見するための分析手法であり，本論文ではこのデータストリームを識別することに焦点を当てる．これはクレジットカードの不正利用識別，POS トランザクションからの優良顧客識別，ネットワークログからの不正侵入識別等，その応用は多岐にわたる．しかし，時間とともに複雑な変化をともなうデータストリームの識別は困難であり，識別性能を向上させることは重要な課題である．

このようなデータストリームの識別は，一般的に教師付き学習が用いられる．従来の教師付き学習は，オンラインで到着するデータを特徴ベクトル $\boldsymbol{x}$ で表し，それを事前に作成した識別器 $y = f(\boldsymbol{x})$ に入力することで，あらかじめ定義されたカテゴリへ識別する．ここで識別精度を上げるために重要なことは，対象データ世界の特徴を抽出することである．従来の識別手法を用いれば，複雑なストリームの特徴をベクトル形式で表現しなければならない．識別で十分な精度が得られないのは，ベクトルの属性値に識別のための特徴が出ていないからである．特徴を出そうと属性の数を増やせば，次元の呪いによって識別器の汎化性が失われ，また，その特徴抽出には深い領域知識が必要である．

†1 公立はこだて未来大学大学院システム情報科学研究科
Graduate School of Systems Information Science, Future University-HAKODATE
†2 公立はこだて未来大学システム情報科学部
School of Systems Information Science, Future University-HAKODATE

一方，線形識別器であるサポートベクタマシン（SVM）は，カーネル法によって非線形識別を行うことができる非線形 SVM へと拡張された．3.1 節で詳細を述べるが，カーネル法は元のデータ空間を高次元空間へ埋め込み，この空間で線形識別を実行する．その計算はカーネル関数と呼ばれる高次元空間上の内積を表す関数で行われ，実際には高次元へ写像することなく，高次元空間上での動作を可能にする．この仕組みはカーネルトリックと呼ばれ，カーネルトリックを用いて非線形識別関数を構成することで非線形識別を可能にする SVM を非線形 SVM という．そしてこの非線形 SVM は，最も識別性能に優れたモデルの 1 つである．また，このカーネル法は近年，構造を持つデータを入力にすることができるように拡張され，これまでに木，配列，グラフ等の構造データに対するカーネル関数が提案されている．このような構造データを用いると，ベクトル形式では失われていたデータの特徴を識別に用いることができ，ベクトル形式で識別するよりも高い識別精度を得ている．

そこで本論文では，カーネル法を拡張して，データストリームを構造データと見なして識別する手法を提案する（**図 1**）．たとえばクレジットカードデータの場合，この構造データは，オンラインで到着するデータ（特徴ベクトル $\boldsymbol{x}$）だけではなく，あるユーザの過去 n 回までの履歴情報をまとめた，1 つのストリーム構造を持つデータ $\boldsymbol{X} = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(n)}\}$ である．また，このストリーム構造データに適用できるカーネル関数は，新しいデータであるほど識別の影響が大きく，古いデータほど識別の影響が小さくなるようにデータを扱い，

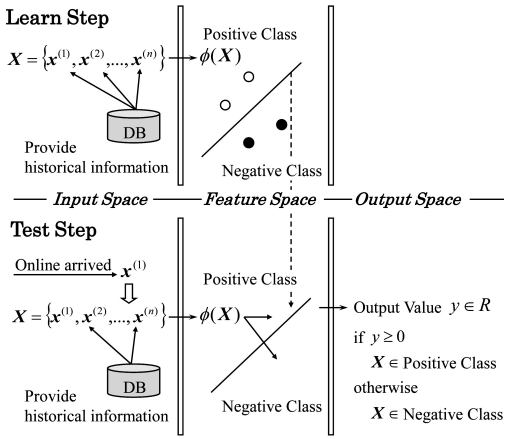


図 1　提案する識別プロセス
Fig. 1　Overview of our proposed classification process.

履歴情報から得られるデータストリームの時間的な変化を特徴にする．

まとめると，既存の手法・研究と比較して，本論文の主張は以下の点である．

- オンラインで到着するデータのみを入力として分析に用いるのではなく，過去 n 回までの履歴をまとめたものを，1 つのストリーム構造を持つデータとして扱い，これを入力にするカーネル関数（本論文ではこれをストリームカーネルと定義する）を構築する．
- 従来のベクトルを入力とした場合（すなわちオンラインで到着したデータのみを用いて，過去の履歴を用いない場合）のカーネルの計算量に対して，このストリームカーネルの計算量は，過去の履歴を遡る件数を n とすると $O(n)$ である．
- 大規模な実際のクレジットカードデータを用いた識別実験を行い，既存の手法よりも優れた精度であることを示す．

本論文では二値の識別に関して述べるが，この制約は重要ではない．なぜなら，どんな二値識別器でも，多重クラスの問題へ拡張することができるからである．たとえば代表的な one-versus-rest は，$C_1, \ldots, C_k$ の k 個のクラスへの識別問題において，あるクラス C_i と，それ以外のクラス $\cup_{j \neq i} C_j$ とを識別する二値識別器を k 個構築する．クラスラベルが未知のデータに対する予測は，k 個の識別器が一貫した結果を出力した場合，そのクラスへ割り当て，そうでない場合は候補の中からランダムにあるいは識別器の出力の中から最もあてはまりが良いクラスへ割り当てられる．

本論文は以下のように構成されている．2 章では関連研究について述べる．3 章では，我々が提案するストリームカーネルの土台となるカーネル法と畳み込みカーネル，そして，ストリームカーネルを適用する識別機械として我々が選択したサポートベクタマシンの説明を行い，4 章で，ストリームカーネルアルゴリズムを導出し，実際のクレジットカードデータを用いた実験の結果を，5 章で示す．最後に，6 章で結言と今後の課題を述べる．

2. 関 連 研 究

大規模に蓄積されたデータストリームを解析し，その結果を有効に活用する研究は，データマイニング[1]，情報検索，統計的学習分野[2] 等において長年活発に行われてきた．カーネル法[3),4)] は，データの非線形構造をとらえる強力な手法として注目されており，分類（clustering）[5] や識別（classification）[6] 等に適用され，高い精度を得ている．

特に識別において，線形識別器である SVM にカーネルトリックを適用し，非線形識別関数を構成することで非線形識別を可能にした非線形 SVM[3] は，最も識別性能に優れたモデルの 1 つである．このことは，我々が行った大規模な実際のクレジットカードデータを不正

利用と正常利用に識別する実験[7] でも示されており，本論文において識別器に非線形 SVM を選択する動機である．

しかし，データストリームを対象にした場合（ネットワーク侵入識別，優良顧客識別等[8]）の識別精度は，静的データを対象にした場合（画像認識，テキスト識別等[9]）と比べると劣り，先行研究では複雑な個人情報ストリームの識別が困難であることを示している．ここで使われている多くの識別アルゴリズム（K-最近傍法，決定木，ベイジアン識別器，ニューラルネットワーク，SVM，ロジスティック回帰等[10]）の入力はベクトル形式であり，過去には識別精度を向上させるために識別に有効なベクトルの属性が考えられてきた[8]．しかし，時間とともに複雑な変化をともなうデータストリームの特徴をすべてベクトルの属性で表現することは困難であり，無理に特徴を出そうとベクトルの属性数を増やせば，次元の呪いによって識別機の汎化性は失われる．

本論文では，従来識別に用いてきたデータだけではなく，過去の履歴データに注目する．そして，遡って得られるデータ群を 1 つのストリーム構造データとして定義し，このストリーム構造データを入力にして識別を行う新しい手法を提案する．まず初めに直面する課題は，どのようにして構造データを入力として扱うかということであるが，上述したカーネル法は近年，配列やグラフ等の構造データを入力にして扱えるように拡張され[12),13)]，テキスト識別等で高い識別精度を得ている[11),14)]．しかし，これらは単に構造データの共通部分構造を数え上げるものであり，次の理由でデータストリームに適用することができない：共通部分構造の数え上げは，部分構造が一致している数をカウントしたものである．たとえば，文字列を配列として扱う文字列カーネルにおいて，2 つの文字列 “cat” と “cart” には，“c, a, t, ca, at, ct, cat” という 7 つの共通部分文字列がある．このようにデータ要素がすべてカテゴリデータであれば，部分構造が一致している数をカウントできる．しかし，データストリームの属性は基本的にカテゴリ属性だけで構成されているわけではなく，連続値の属性も含む．したがって，共通部分構造の数え上げという概念を持ち込むことができない．また，従来手法はデータストリームの時間的に変化する性質を考慮していない．

これらの課題から，本論文では構造データを扱う基礎理論である畳み込みカーネルを用いて，データストリームに適用できるカーネル法を提案する．提案手法は，新しいデータほど識別の影響を大きく，古いデータほど識別の影響を小さくするようにデータを扱い，識別を行う．これにより，ストリームの時間的な変化を特徴にすることができる．また，上述した配列・グラフ等の構造データに対するカーネル法が高い精度を得ていることは，本研究でデータストリームに適用できるカーネル法を提案することで精度を向上させる根拠となる．

3. カーネル法と非線形サポートベクタマシン

3.1 カーネル法

カーネル法は，データの非線形構造をとらえる強力な手法であり，2 つの本質的な要素からなる．

(1)　データを高次元特徴空間に埋め込む．

(2)　高次元特徴空間で，線形識別アルゴリズムを適用する．

線形識別アルゴリズムは多く研究されているが，真の識別境界が非線形である場合，十分な性能が実現できない．そこで，カーネル法は元のデータ空間をいったん線形識別アルゴリズムを適用できる高次元空間へ写像し，この空間で線形識別アルゴリズムを適用する（**図 2**）．しかし，カーネル法はデータを高次元特徴空間では明確に示さず，カーネル関数と呼ばれる半正定値関数 $K : \mathcal{X} \times \mathcal{X} \to R$ を用いて，高次元特徴空間での動作を可能にする．

$$K(\boldsymbol{x}, \boldsymbol{z}) = \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{z}) \rangle \tag{1}$$

Mercer の定理を満たす半正定値関数 K をカーネル関数と呼び，これは ϕ によって写像された高次元特徴空間上での内積を意味する．このように実際に高次元特徴空間へ写像する計算を避け，入力空間でのカーネル関数を用いて高次元特徴空間上での内積を計算する仕掛けをカーネルトリックと呼ぶ．また，カーネル関数には次のようなものが知られている．

- 多項式カーネル

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i^T \boldsymbol{x}_j + c)^d \tag{2}$$

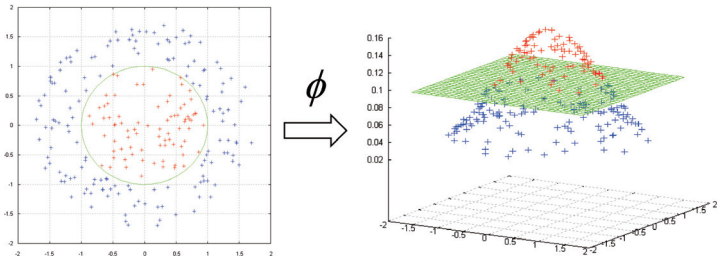


図 2　カーネル法による線形分離
Fig. 2　Linear separation using the kernel methods.

- ガウスカーネル

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{1}{\sigma^2}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2\right) \tag{3}$$

- シグモイドカーネル

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \tanh(a\boldsymbol{x}_i^T\boldsymbol{x}_j - b) \tag{4}$$

線形識別器である SVM は，上述したカーネルトリックを用いて非線形識別を行うことができる識別器へと拡張された．SVM による識別は，異なるクラスのデータ間の距離（マージン）を最大にする超平面を考える．このときの目的関数と識別関数は，特徴ベクトルの内積のみに依存した形で記述される．たとえば識別関数は，クラスラベル変数 t，サポートベクタの集合 S，Lagrange 乗数 $\alpha_i^* > 0$，最適な閾値 h^* を用いて次式で書ける．

$$f(\boldsymbol{x}) = \sum_{i \in S} \alpha_i^* t_i \boldsymbol{x}_i^T \boldsymbol{x} - h^* \tag{5}$$

この線形識別関数は，カーネルトリックを適用することにより以下の形に書き変えることができる．

$$f(\boldsymbol{x}) = \sum_{i \in S} \alpha_i^* t_i \phi(\boldsymbol{x}_i)^T \phi(\boldsymbol{x}) - h^* \tag{6}$$

$$= \sum_{i \in S} \alpha_i^* t_i K(\boldsymbol{x}_i, \boldsymbol{x}) - h^* \tag{7}$$

ここで，元の空間よりも高次元に写像するカーネル関数 K を選択すれば，高次元空間上で線形 SVM を実行したことになり，これは元の空間で非線形識別を実行したことと等価である（図 2）．このように，カーネルトリックによって非線形識別関数を構成することで，非線形識別を可能にした SVM を，非線形 SVM という．

また，これらカーネル関数は 2 つの入力の類似度を定めていると考えることができる．すなわち，適当に類似度 R を定義した関数 K を，カーネル関数として使うことができる．したがって，上述したカーネル関数の 2 つの入力 $\boldsymbol{x}_i$，$\boldsymbol{x}_j$ がベクトルではなく，DNA 配列やテキスト等の配列，XML や HTML で記述された構文解析木のような木構造，あるいは化合物の分子構造のようなグラフで表現される場合にも，カーネル法を適用することができる．本研究で提案するのはストリーム構造を持ったデータ間のカーネル関数であり，これら構造を持ったデータ間のカーネルを定義するための基礎理論である畳み込みカーネルについて，次節で述べる．

3.2　畳み込みカーネル

本節では，カーネル関数の入力がベクトルではなく，配列やグラフ等の構造を持ったデータに対するカーネル関数を構築するための基礎理論である畳み込みカーネルについて述べる．畳み込みカーネル[15] は，構造を持つデータ間のカーネル関数はその構造の部分構造どうしのカーネル関数によって再帰的に計算されるという考えに基づいている．

$\boldsymbol{x}$ はある構造を持つデータ，$S(\boldsymbol{x})$ は $\boldsymbol{x}$ に含まれる部分構造の集合，s_x をその部分構造とする．これは $\boldsymbol{z}$ についても同様である．K_s を部分構造間のカーネル関数とすると，畳み込みカーネルは以下で表現される．

$$K(\boldsymbol{x}, \boldsymbol{z}) = \sum_{s_x \in S(\boldsymbol{x})} \sum_{s_z \in S(\boldsymbol{z})} K_s(s_x, s_z) \tag{8}$$

また，畳み込みカーネルは部分構造 s に対する重み $f(s|\boldsymbol{x})$ を用いて次のように記述することができる．

$$K(\boldsymbol{x}, \boldsymbol{z}) = \sum_{s_x \in S(\boldsymbol{x})} \sum_{s_z \in S(\boldsymbol{z})} f(s_x|\boldsymbol{x}) f(s_z|\boldsymbol{z}) K_s(s_x, s_z) \tag{9}$$

この部分構造のカーネル関数 K_s は，さらに分けられた部分構造による畳み込みカーネルにより，再帰的に表現される．つまり，構造を持ったデータ $\boldsymbol{x}$ と $\boldsymbol{z}$ は，ここから取り出される部分構造 $s_x \in S(\boldsymbol{x})$ と $s_z \in S(\boldsymbol{z})$ のカーネル関数の値をすべて足し合わせることで全体のカーネル関数が定義される．本論文では 4 章でストリーム構造を持つデータ $\boldsymbol{X} = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(n)}\}$ 間のカーネル関数を提案する．このカーネル関数を我々はストリームカーネルと呼ぶが，ストリームカーネルも畳み込みカーネルを基礎理論としている．

4.　ストリームカーネル

4.1　ストリーム構造データの定義

データストリームに対する従来の識別手法は，オンラインで到着したデータをベクトル $\boldsymbol{x} = (x_1, x_2, \ldots, x_d)$ で表現し，これを入力に用いる．これに対し，提案する識別手法は，次の特徴を持った構造データを入力に用いる．

(1)　オンラインで到着した $\boldsymbol{x}^{(1)}$ だけではなく，過去の履歴 n 個のデータを持つ．

(2)　n 個の前後のデータ間に，$n-1$ 個の時間間隔を持つ．

本論文ではこれをストリーム構造データと定義し，次のように表現する．

$$\boldsymbol{X} = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(n)}\} \tag{10}$$

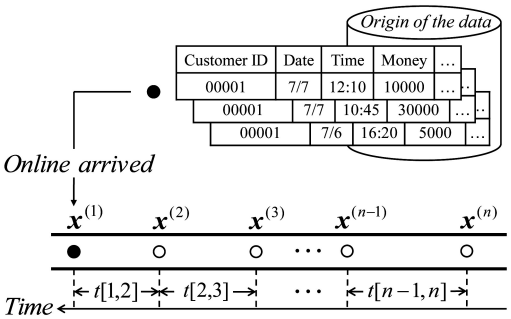


図 3　ストリーム構造データ
Fig. 3　Structured data of the data stream.

なお，$\boldsymbol{x}^{(1)}$ が新しいデータ，$\boldsymbol{x}^{(n)}$ が古いデータとし，括弧内の上付き添え字を履歴番号と呼ぶ．時間間隔とは，データが到着した時間の間隔であり，たとえば $\boldsymbol{x}^{(1)}$ と $\boldsymbol{x}^{(2)}$ の時間間隔は $t[1,2]$ と表す（**図 3**）．

ここで，時間間隔を計算に用いるうえで注意すべきことがある．時間間隔は，古い情報ほど識別の影響を小さくする重みを計算するときに用いられるが（4.3 節参照），このとき，遡った期間全体を 1 に正規化する．つまり，たとえば $n = 4$ とし，過去の履歴 4 件を遡る場合，得られる時間間隔は，$t[1,2]$，$t[2,3]$，$t[3,4]$ であり，その合計を $t[1,2]+t[2,3]+t[3,4] = 1$ となるように値を変換する．これは単に，各時間間隔についての，遡った期間全体に対する比を考えるだけである．たとえばクレジットカードデータを考えると，クレジットカードを 1 週間に 1 度の間隔で使う人もいれば，1 カ月に 1 度の間隔で使う人もいる．時間間隔を正規化することで，各人それぞれの利用の間隔を同じ尺度でとらえることができる．

4.2　部分構造間のカーネル

本論文で提案するストリームカーネルは，3.2 節で述べた畳み込みカーネルが基本理論となっている．したがって，まず 4.1 節で定義したストリーム構造データ $\boldsymbol{X} = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(n)}\}$ の部分構造 s を定義する．例として，遡る履歴の数を $n = 3$ とした $\boldsymbol{X} = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \boldsymbol{x}^{(3)}\}$ の場合について考える．$\boldsymbol{X}$ の部分構造全体の集合を $S(\boldsymbol{X})$ とすると，順序関係を考慮して，$S(\boldsymbol{X})$ は以下の 6 つの部分構造へ分割される．このとき，遡る履歴の数 n に対して，得られる部分構造の数は $n(n+1)/2$ である．

$$\begin{aligned} S(\boldsymbol{X}) &= \{s_1, s_2, s_3, s_4, s_5, s_6\} \\ &= \{\{\boldsymbol{x}^{(1)}\}, \{\boldsymbol{x}^{(2)}\}, \{\boldsymbol{x}^{(3)}\}, \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}\}, \{\boldsymbol{x}^{(2)}, \boldsymbol{x}^{(3)}\}, \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \boldsymbol{x}^{(3)}\}\} \end{aligned}$$

次に，分割された部分構造間のカーネルの組合せを考える．ここで，異なる履歴番号で観測されたデータが独立であるとする．すなわち，$K(\boldsymbol{x}^{(i)}, \boldsymbol{z}^{(j)}) = 0$，$i \neq j$ とし，部分構造間のカーネルを，部分構造に含まれるデータのカーネルの積で表現する．すると，部分構造のデータ数が同じで，かつ履歴番号が同じであるときのみ，その部分構造どうしのカーネル K_s は計算され，次式で表される．

$$K_s(s_x, s_z) = \prod_{i=1}^{n} K(\boldsymbol{x}^{(i)}, \boldsymbol{z}^{(i)}) \tag{11}$$

$$K(\boldsymbol{x}^{(i)}, \boldsymbol{z}^{(i)}) = \begin{cases} K(\boldsymbol{x}^{(i)}, \boldsymbol{z}^{(i)}) & \text{if } \boldsymbol{x}^{(i)} \text{ and } \boldsymbol{z}^{(i)} \text{ exist} \\ 0 & \text{if } \boldsymbol{x}^{(i)} \text{ or } \boldsymbol{z}^{(i)} \text{ exist} \\ 1 & \text{otherwise} \end{cases}$$

なお，$K(\boldsymbol{x}^{(i)}, \boldsymbol{z}^{(i)})$ は 3.1 節で述べた半正定値カーネルである．これにより，考えるべきカーネルの組合せは $n(n+1)/2$ 通りとなる．

4.3　部分構造の重み

本節では，畳み込みカーネルにおける部分構造の重み $f(s|X)$ を定義する．この $f(s|X)$ は，4.2 節で定義した部分構造のカーネルが，全体の類似度にどれほどの影響力を持つかを表す．これを考えるうえで，理解が単純になるよう再度クレジットカードデータの例を用いる．

オンラインで到着した顧客 A のデータを $\boldsymbol{x}_A^{(1)}$ とすると，この利用 $\boldsymbol{x}_A^{(1)}$ が正常利用か，不正利用かを識別するために，顧客 A の利用履歴を過去 n 回遡ったストリーム構造データ $\boldsymbol{X}_A = \{\boldsymbol{x}_A^{(1)}, \boldsymbol{x}_A^{(2)}, \ldots, \boldsymbol{x}_A^{(n)}\}$ を得る．$\boldsymbol{x}_A^{(1)}$ が新しいデータ，$\boldsymbol{x}_A^{(n)}$ を古いデータとすると，$\boldsymbol{x}_A^{(1)}$ と $\boldsymbol{x}_A^{(n)}$ を同じ重みで扱うと正確な識別を行うことができない．なぜなら，知りたいのは到着したデータ $\boldsymbol{x}_A^{(1)}$ が不正利用か，正常利用かということである．古いデータがいくら正常利用らしくても，到着したデータが明らかに不正利用らしければ，到着したデータの方を優先すべきである．したがって，新しいデータを含む部分構造であるほど重み（影響度）を大きく，古いデータを含む部分構造であるほど重みを小さくする．この重み $f(s|X)$ は，4.1 節で定義した時間間隔 t と，新たに導入するパラメータ $\lambda \in (0,1)$ を用いた単調減少関数 λ^t によって定義される．

ここで，部分構造に含まれるそれぞれのデータに対して具体的に重みを与える．すなわち n 番目の履歴データに対して，重みを $\prod_{i=1}^{n-1} \lambda^{t[i,i+1]}$ と定義する．例として，$s_x = \left\{\boldsymbol{x}^{(2)}, \boldsymbol{x}^{(3)}, \boldsymbol{x}^{(4)}\right\}$ と，$s_z = \left\{\boldsymbol{z}^{(2)}, \boldsymbol{z}^{(3)}, \boldsymbol{z}^{(4)}\right\}$ の重み $f(s_x|X)$ と $f(s_x|Z)$ を求める．部分構

表 1　部分構造 s_x に含まれるデータの重み
Table 1　Weight of data in substructure s_x.

	$\boldsymbol{x}^{(2)}$	$\boldsymbol{x}^{(3)}$	$\boldsymbol{x}^{(4)}$
重み	$\lambda^{t_x[1,2]}$	$\lambda^{t_x[1,3]}$	$\lambda^{t_x[1,4]}$

造 s_x に含まれるデータそれぞれの重みを**表 1** で示す．なお，t_x は，ストリーム構造データ X が持つ時間間隔であるとし，記述の簡略化のため，$t[1,m]=\sum_{i=1}^{m-1} t[i,i+1]$ とする．この $\boldsymbol{x}$ をすべて $\boldsymbol{z}$ で置き換えれば，同様に部分構造 s_z に含まれるデータそれぞれの重みを表す．簡略化せずに重みを詳細に記述すれば，$f(s_x|X)$ と $f(s_z|Z)$ はそれぞれ次式で表される．

$$f(s_x|X)=\lambda^{t_x[1,2]}\lambda^{t_x[1,2]+t_x[2,3]}\lambda^{t_x[1,2]+t_x[2,3]+t_x[3,4]} \tag{12}$$
$$f(s_z|Z)=\lambda^{t_z[1,2]}\lambda^{t_z[1,2]+t_z[2,3]}\lambda^{t_z[1,2]+t_z[2,3]+t_z[3,4]} \tag{13}$$

したがって，このときの部分構造の重みを含めたカーネルは，式 (11) を用いて，

$$\begin{aligned} f(s_x|X)f(s_z|Z)K_s(s_x,s_z)=\lambda^{3t_x[1,2]+2t_x[2,3]+t_x[3,4]}\lambda^{3t_z[1,2]+2t_z[2,3]+t_z[3,4]} \\ \times K(\boldsymbol{x}^{(2)},\boldsymbol{z}^{(2)})K(\boldsymbol{x}^{(3)},\boldsymbol{z}^{(3)})K(\boldsymbol{x}^{(4)},\boldsymbol{z}^{(4)}) \end{aligned} \tag{14}$$

となる．

4.4　ストリームカーネル計算アルゴリズム

$K_n(\boldsymbol{X},\boldsymbol{Z})$ を，$\boldsymbol{X}$ と $\boldsymbol{Z}$ がともに過去 n 件のストリーム構造データからなるときのストリームカーネルとする．また，過去 n 件からなる $\boldsymbol{X}$ に，さらにもう 1 件遡って得られる $\boldsymbol{x}^{(n+1)}$ が，$\boldsymbol{X}$ に付与されることを $\boldsymbol{X}\boldsymbol{x}^{(n+1)}$ と表現する．もちろんこれは $\boldsymbol{Z}$ についても同様であるが，以降では $\boldsymbol{Z}$ にも同じ作業を与えたとして，その記述は省略する．

ここで具体的に，例として $n=1$ の場合から考える．このとき，$\boldsymbol{X}$ は $\boldsymbol{X}=\{\boldsymbol{x}^{(1)}\}$ であり，$S(\boldsymbol{X})=\{s_1\}=\{\boldsymbol{x}^{(1)}\}$ である．したがって，単純に $K_1(\boldsymbol{X},\boldsymbol{Z})$ は次で表される．

$$K_1(\boldsymbol{X},\boldsymbol{Z})=K(\boldsymbol{x}^{(1)},\boldsymbol{z}^{(1)}) \tag{15}$$

次に，$\boldsymbol{X}=\{\boldsymbol{x}^{(1)}\}$ に $\boldsymbol{x}^{(2)}$ が付与され，$\boldsymbol{X}\boldsymbol{x}^{(2)}=\{\boldsymbol{x}^{(1)},\boldsymbol{x}^{(2)}\}$ となったときの $K_2(\boldsymbol{X}\boldsymbol{x}^{(2)},\boldsymbol{Z}\boldsymbol{z}^{(2)})$ を考える．$\boldsymbol{X}\boldsymbol{x}^{(2)}=\{\boldsymbol{x}^{(1)},\boldsymbol{x}^{(2)}\}$ により，4.2 節と同様の手法で，$\boldsymbol{X}\boldsymbol{x}^{(2)}$ は 3 つの部分構造に分割できる．

$$\begin{aligned} S(\boldsymbol{X}\boldsymbol{x}^{(2)})&=\{s_1,s_2,s_3\}\\ &=\{\{\boldsymbol{x}^{(1)}\},\{\boldsymbol{x}^{(2)}\},\{\boldsymbol{x}^{(1)},\boldsymbol{x}^{(2)}\}\} \end{aligned}$$

式 (9)，(11)，(14) により，$K_2(\boldsymbol{X}\boldsymbol{x}^{(2)},\boldsymbol{Z}\boldsymbol{z}^{(2)})$ は次式で記述できる．なお数式の簡略化のため，$t_x[i,i+1]+t_z[i,i+1]=T[i,i+1]$ とする．

$$\begin{aligned} K_2(\boldsymbol{X}\boldsymbol{x}^{(2)},\boldsymbol{Z}\boldsymbol{z}^{(2)})&=K(\boldsymbol{x}^{(1)},\boldsymbol{z}^{(1)})+\lambda^{T[1,2]}K(\boldsymbol{x}^{(2)},\boldsymbol{z}^{(2)})\\ &\quad+\lambda^{T[1,2]}K(\boldsymbol{x}^{(1)},\boldsymbol{z}^{(1)})K(\boldsymbol{x}^{(2)},\boldsymbol{z}^{(2)}) \end{aligned} \tag{16}$$

ここで，式 (15) と式 (16) を比較し，$\boldsymbol{x}^{(2)}$ が付与されることで増加するカーネルの量を計算する．この量を J_2 とすると，J_2 は次のように J_1 を用いた形で書くことができる．

$$\begin{aligned} J_2&=K_2(\boldsymbol{X}\boldsymbol{x}^{(2)},\boldsymbol{Z}\boldsymbol{z}^{(2)})-K_1(\boldsymbol{X},\boldsymbol{Z})\\ &=\lambda^{T[1,2]}K(\boldsymbol{x}^{(2)},\boldsymbol{z}^{(2)})+\lambda^{T[1,2]}K(\boldsymbol{x}^{(1)},\boldsymbol{z}^{(1)})K(\boldsymbol{x}^{(2)},\boldsymbol{z}^{(2)})\\ &=\lambda^{T[1,2]}K(\boldsymbol{x}^{(2)},\boldsymbol{z}^{(2)})\{1+K(\boldsymbol{x}^{(1)},\boldsymbol{z}^{(1)})\}\\ &=\lambda^{T[1,2]}K(\boldsymbol{x}^{(2)},\boldsymbol{z}^{(2)})\{1+J_1\} \end{aligned} \tag{17}$$

同様に，$\boldsymbol{X}=\{\boldsymbol{x}^{(1)},\boldsymbol{x}^{(2)}\}$ に $\boldsymbol{x}^{(3)}$ を付与した場合を展開しても，$\boldsymbol{x}^{(3)}$ が付与されることで増加するカーネルの量 J_3 もまた，J_2 を用いた形で書くことができる．したがって，遡るデータが 1 つ増えることで，全体のカーネルに加えられる部分構造データのカーネルの量を J_n とすると，ストリームカーネルは以下のように一般化した形で，再帰的に表現することができる．

$$K_n(\boldsymbol{X}\boldsymbol{x}^{(\boldsymbol{n})},\boldsymbol{Z}\boldsymbol{z}^{(\boldsymbol{n})})=K_{n-1}(\boldsymbol{X},\boldsymbol{Z})+J_n \tag{18}$$
$$J_n=(1+J_{n-1})K(\boldsymbol{x}^{(n)},\boldsymbol{z}^{(n)})\prod_{i=1}^{n-1}\lambda^{T[i,i+1]} \tag{19}$$

ただし，$K_0(\boldsymbol{X},\boldsymbol{Z})=1$，$J_1=K(\boldsymbol{x}^{(1)},\boldsymbol{z}^{(1)})$ である．動的計画法を用いることにより，この計算量は遡るデータ数 n に対して $O(n)$ で済む．また，$\boldsymbol{X}$ と $\boldsymbol{Z}$ のデータ数が異なる場合，このままでは，データ数が少ない部分集合のカーネルのみを考えてしまう．たとえば，$\boldsymbol{X}=\{\boldsymbol{x}^{(1)},\boldsymbol{x}^{(2)},\boldsymbol{x}^{(3)}\}$，$\boldsymbol{Z}=\{\boldsymbol{z}^{(1)},\boldsymbol{z}^{(2)}\}$ とすると，$\boldsymbol{x}^{(3)}$ はカーネルの計算にまったく用いられない．そこで，以下の式で正規化を行い．これをストリームカーネルの最終出力値とする．

$$K(\boldsymbol{X},\boldsymbol{Z})=\frac{K_n(\boldsymbol{X},\boldsymbol{Z})}{\sqrt{K_n(\boldsymbol{X},\boldsymbol{X})K_n(\boldsymbol{Z},\boldsymbol{Z})}} \tag{20}$$

さらに，このストリームカーネルが半正定値性を満たすことを述べる．いくつかの半正定値性を満たすカーネル関数は，これを足し合わせたり，掛け合わせたり，また定数倍したり

したものもまた，半正定値性を満たすカーネル関数となることが知られている．ストリームカーネルは，半正定値性を満たすカーネル関数を基本として用い，部分構造どうしのカーネルの計算ではこのカーネルを掛け合わせ，部分構造の重みはカーネルを定数倍し，畳み込みで表現された構造全体のカーネルはこれらを足し合わせたものである．したがって，ストリームカーネルは半正定値性を満たす．

5. 実験評価

本実験では，実際のデータストリームであるクレジットカードデータ（実験データの詳細は 5.2 節参照）に対して，ガウスカーネルを用いた通常の SVM と，提案するストリームカーネル SVM（以下 SKSVM と記す）を適用し，正常利用と不正利用に識別した．また，その性能を以下の点から比較した：

(a)　学習・検証にかかる時間
(b)　識別の精度（正答率）
(c)　モデルの性能（CAP 曲線）

5.1 実験準備

実験は 2 GHz の CPU と，3 GB のメモリを持つハードウェア環境を使用した．提案する SKSVM は，OSS である svm-light [16]（C 言語）上で実装を行った．

実験で使用したカーネル関数は，ガウスカーネル $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/\sigma^2\right)$ である．ガウスカーネルは，パラメータ σ の設定によってはシグモイドカーネルのようにも動作し，また，先行研究[7] で行った実験結果により，ガウスカーネルが識別に最も有効に動作すると判断した．ガウスカーネルのパラメータ σ は，事前に $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2$ の統計をとり，これらが適切に $K(\boldsymbol{x}_i, \boldsymbol{x}_j) \in (0, 1)$ を得るような範囲を確定し，SVM の正則化パラメータ C とともに試行錯誤的に決定した．

ストリームカーネルのパラメータ λ は，一番最近のデータのカーネル関数の値が，ストリームカーネル全体の影響の 7 割程度を占めるように設定した．これは，過去の情報の重みを大きくしすぎてしまうと，一番重要な最近のデータの特徴が埋もれてしまうためである．また，顧客の遡る履歴数 n は，$n = 5$ に設定した．これら実験で用いたパラメータを**表 2** で示す．

本実験ではモデルの性能を CAP 曲線によって評価するために，新たにスコア値という考えを導入している．スコア値とは，値が高いものほど不正利用であるらしく，逆に値が低いものほど正常利用であるらしいという指標である．具体的には，SVM の出力 y に対して，

表 2　実験で用いたパラメータ
Table 2　The parameters used in the experiment.

	σ	C	n	λ
SVM	0.05	0.5	—	—
SKSVM	0.05	0.3	5	0.1

スコア値 *score* は以下の式で 1〜1000 の値で出力される．

$$score = 1000 \times \frac{1}{1 + e^{-y}} \tag{21}$$

5.2 実験データ

本実験で用いたデータセットは，実際のクレジットカードデータである．クレジットカードデータは，約 2 年分のデータが用意されており，そのデータ量は約 1 TB に及ぶ．クレジットカードデータの属性は，大きく次の 2 つのグループに分類される．

(a)　オーソリデータ属性：　利用時の状況を記述した属性
(b)　振舞いデータ属性：　顧客の行動パターンを記述した属性

属性（a）は，単にクレジットカードの取引データ（以下オーソリデータと記す）の属性である．たとえばこの属性には，顧客 ID，生年月日，利用時間，利用金額，購入商品コード等があり，計 84 属性からなる．しかし，これらの属性には識別に有効な（不正利用と正常利用で，はっきり値が異なるような）属性は少ない．したがって，この属性だけを用いて識別アルゴリズムを適用しても高い精度は得られない．そこで，識別の特徴として貢献するような以下の属性（b）を，人工的に作成する．

属性（b）は過去の履歴情報から作成した顧客モデル（過去の利用金額の平均や分散，過去の利用時間帯の頻度等）との乖離値であり，顧客の行動パターンを記述した属性である．たとえばある人が高額な買い物をしたとする．その人が前回も同程度の額の買い物をしていたならば，その利用は本人，すなわち正常利用らしいが，逆に前回低額な買い物をしていたならば，第三者利用，すなわち不正利用である可能性があると判断することができる（**図 4**）．このように顧客モデルに基づいてその顧客の行動パターンの時間的変化に注目することは，クレジットカードの領域だけではなく，ネットワーク侵入や個人認証等，他の多くの領域・データにあてはまり，有用である．この属性（b）には「前回利用時間との差」，「前回利用金額との差」，「過去 6 カ月の曜日ごとの利用回数との比較」等があり，計 40 属性からなる．本実験で用いたクレジットカードデータセットは属性（a）と属性（b）から 55 属性を分析に用いた．データセットの詳細を**表 3** に示す．

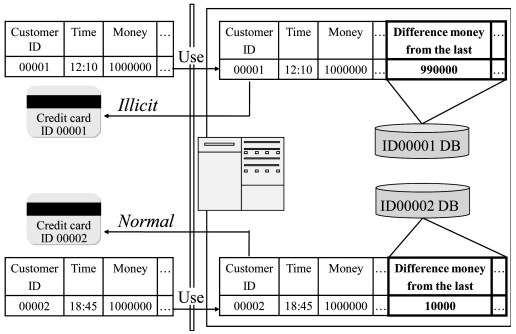


図 4　振舞い属性を考慮した識別：「前回利用金額との差」という属性を加えることで，特徴ベクトルに不正らしさが現れている
Fig. 4　Classification considered the behavior attribute: Suspicious appears by added “difference money from the last” in feature vector.

表 3　実験データ
Table 3　Credit card dataset in our experiment.

	データサイズ	総レコード数	不正利用レコード数	正常利用レコード数
SVM 学習データ	225 MB	388,611	4,536	384,075
SKSVM 学習データ	1,353 MB	388,611	4,536	384,075
SVM 検証データ	418 MB	720,920	202	720,718
SKSVM 検証データ	2 GB	720,920	202	720,718

SKSVM と SVM の学習・検証データの違いは，利用時点から過去 n 回の履歴があるか，ないかである．SKSVM の学習・検証データは，**図 5** で示すように，上述した顧客モデルを更新しながら属性（b）を作成して，ストリーム構造データを作成する．本実験では，$n = 5$ とし過去 5 回の履歴を遡ってストリーム構造データを作成し，識別を行う．

クレジットカードデータを，ストリーム構造データとして扱ううえでの注意すべき点として，顧客のマスタ情報を繰り返し用いることがあげられる．たとえば，顧客の「性別」という属性を分析に用いるとする．すると，過去 2 件目，3 件目にも同じ性別の属性が，繰り返し入ることになる．一見これはデータの冗長性から無駄のように見えるが，これらはすべて必要である．なぜなら，冗長性のある「性別」という属性を 1 件目だけに用いて，それ以降の過去 2 件目，3 件目には用いない場合を考える．そしてもし，その「性別」という属性が，不正利用につながる重要な属性であるとすれば，部分集合間のカーネル（4.2 節参照）

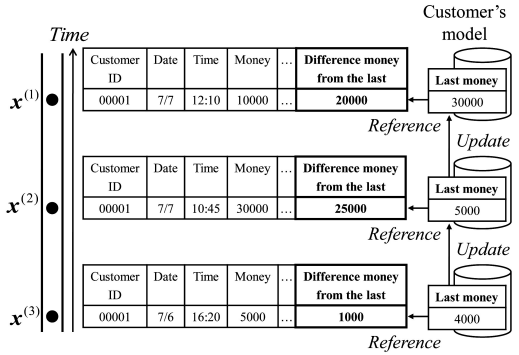


図 5　ストリーム構造データの作成
Fig. 5　The preparations for structured data of data stream.

表 4　計算時間の比較
Table 4　Comparison of running time.

	SVM	SKSVM
学習時間	2,839.91	21,153.78
検証時間	1,936.32	21,166.69

を考える際，不十分な情報で過去の履歴を処理してしまい，全体として，特徴が薄れてしまうためである．

5.3　実験結果と考察

5.3.1　学習と検証の時間

通常の SVM と，SKSVM の学習時間・検証時間の比較を**表 4** で示す．実験の結果，学習時間・検証時間ともに，提案手法は約 10 倍の時間がかかった．SKSVM のカーネルの計算量は，通常の SVM のカーネルの計算量に比べ，理論上では遡る履歴の数 n に対して $O(n)$ で増加するものの，実際のデータでは欠損値の場合の処理，時間間隔データの変換（4.1 節参照）等の事前計算処理が必要であるため，表 4 の結果はこれらオーバヘッドを含んだものである．

次に，通常の SVM と，SKSVM のモデルの大きさであるサポートベクタ（SV）の数の比較を**表 5** で示す．これは式 (7) において検証の際に計算を繰り返す数を表している．したがって，SV の数が少なければ，より少ない計算量で検証を行うことができることを示し，また，スパースな解が得られていることになる．表 5 で示すように，SKSVM の方がより

表 5　サポートベクタ（SV）の数の比較
Table 5　Comparison of the number of Support Vector (SV).

	SVM	SKSVM
SV の数	8,438	7,844

表 6　SVM の識別結果
Table 6　Experimental result of SVM.

	SVM が不正利用と判断	SVM が正常利用と判断	合計
実際は不正利用	30	172	202
実際は正常利用	1,080	719,638	720,718
合計	1,110	719,810	720,920

正答率：99.82%　（正識別：719,668　誤識別：1,252　合計：720,920）

表 7　SKSVM の識別結果
Table 7　Experimental result of SKSVM.

	SKSVM が不正利用と判断	SKSVM が正常利用と判断	合計
実際は不正利用	28	174	202
実際は正常利用	561	720,157	720,718
合計	589	720,331	720,920

正答率：99.89%　（正識別：720,185　誤識別：735　合計：720,920）

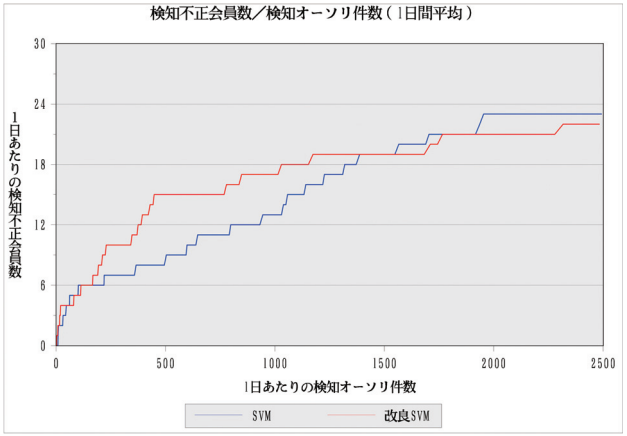


図 6　CAP 曲線による性能比較：CAP 曲線はモデルの識別性能を表す．SVM によって予測したスコア値（5.1 節参照）を高い方から順に並べ，度数分布で描いたグラフが CAP 曲線である．この曲線は図 7 で示すように，A に近いほど判別能力の高いモデルであり，C に近いほど判別能力の低いモデルであるといえる
Fig. 6　Accuracy comparison by CAP curve: CAP curve represents accuracy of the model. Arranged scores (see 5.1) in descending order, the curve is the frequency distribution of illegal users. Following Fig. 7, we are able to see that the closer the curve is to A, the better accuracy, in contrast, the closer the curve is to C, the poorer accuracy.

スパースな解を得ていることが分かる．

5.3.2　識別の精度（正答率）

識別の精度を評価する指標には，一般的に適合率（precision）と再現率（recall）が用いられる．しかし，本実験で用いた実際のクレジットカードデータは，不正利用の数が正常利用の数に比べて約 0.02%と極端に少なく，また，識別能力の弱い識別器を用いるとすべてを正常利用であると判断してしまうほど，正常利用と不正利用の識別は困難である．したがって，適合率と再現率はともに数%という値しかとることができず，実際のクレジットカードデータを不正利用と正常利用に識別するというこの領域においては，これらの指標は適当ではない．提案手法である SKSVM による識別精度の向上を明確にするため，本実験の識別精度は正答率で評価する．正答率は，識別を行った全件（720,920 件）に対し，その識別結果が正しかった件数の割合である．**表 6** と**表 7** で示すように，SVM の正答率が 99.82%であったのに対し，SKSVM の正答率は 99.89%であり，0.07%向上している．

割合からはそれほどの向上は見られないが，これは大きな変化である．表 6 において，SVM が不正利用であると判断した件数は，1,110 件である．しかし，そのうち 1,080 件は実際には正常利用であり，これは間違った識別である．一方，表 7 において，SKSVM が不正利用であると判断した件数は，586 件である．これは SVM と比べると約半分のヒット件数であるにもかかわらず，実際に不正利用であった件数は変わらない（30 件と 28 件）．今回検証データとして用いたデータセットは不正利用の数が少ないため，この変化は割合として大きく現れないが，実際には SKSVM は不正利用データを発見するときのノイズを半分近く減らしていることが分かる．

5.3.3　CAP 曲線による比較

CAP（Cumulative Accuracy Profiles）曲線は，予測モデルの性能を評価するのに用いられ，実際のクレジットカード業界でも使用されている．横軸は，不正利用である確率（スコア値（5.1 節参照））が高いと識別器が判断したクレジットカードデータを順に並べた数であり，縦軸は，その中で実際に不正利用であった顧客の人数である．**図 6** は，通常の SVM と，SKSVM の CAP 曲線による比較を示している．もし得られた曲線が，**図 7** で示す C

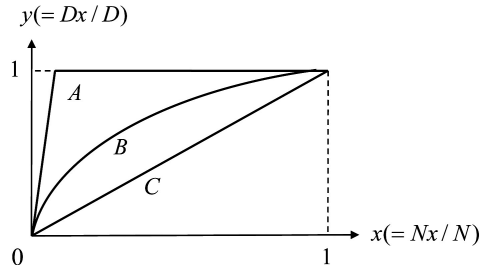


図 7　CAP 曲線の見方：全データ数を N とし，その中のすべての不正利用の数を D とする．全データをスコア値（5.1 節参照）の高い方から順に並べ，上位 N_x 件を抽出したとき，その中にある不正利用の数を D_x とする．このとき $x = N_x/N$，$y = D_x/D$ として描いた曲線が CAP 曲線である

Fig. 7　Detail of CAP curve: N is the number of the data, D is the number of illegal users in N. Arranged scores (see 5.1) in descending order, we derive the top N_x from N. D_x is a number of illegal users in N_x. Then CAP curve is described with $x = N_x/N$, $y = D_x/D$.

に近いならば，スコア値はランダムに付けられたものと等価であり，モデルにまったく説明力がないことを示す．対称的に，得られた曲線が A に近いならば，実際に不正利用であるクレジットカードデータに対して高いスコア値を付けられていることになり，モデルの性能が高いことを示す．実際には A と C の間の，B のような曲線をとる．

図 6 から，SKSVM は不正らしい上位 500 件を監視すれば，15 人の不正利用者を検知できていることを示している．これに対し，SVM は同様に不正らしい上位 500 件を監視しても，約半数の 8 人しか検知できていない．さらに，SKSVM が不正らしい上位 500 件を監視するだけで検知できる 15 人の不正利用者を，SVM が検知しようとすれば，不正らしい上位 1,000 件（SKSVM の倍）もの監視をしなければならないということも示している．

6. おわりに

本論文では，従来の識別手法を用いては高い精度を得ることが困難なデータストリームに対して，カーネル法を拡張した新しい識別の手法を提案した．これは，過去複数のデータを 1 つのストリーム構造データとして識別するものであり，ストリームカーネルという新しいカーネル関数を導入した．また，実際の大規模なクレジットカードデータを用いて，正常利用と不正利用に識別する実験を行い，提案手法は従来手法に比べ誤識別の数を約 40%減少させ，不正利用らしさが上位のデータ中では最大で約 2 倍の不正利用顧客数を検出した．さらに，本実験で設定したパラメータ（遡る件数 n と，過去の影響度合 λ）は，試験的に設定したものであり，今後これらを調整することでさらなる精度が得られると考える．

今後の課題として，まず処理時間の短縮があげられるが，これは能動学習[6] によって達成できると考える．次に，提案手法はカーネル関数のパラメータのほかに，「遡る履歴数 n」と「過去の影響度合 λ」を設定する必要がある．これは試行錯誤によって決定するしかない．さらに，本研究の成果はカーネル法を用いた多くのデータマイニング・機械学習の手法に導入することができるため，様々な実データを用いた実証実験により，提案手法の有効性を検証する必要がある．

謝辞　本研究の実験で使用したクレジットカードのデータの提供，および実験に対するアドバイスをいただいた（株）インテリジェントウェイブの関係者の方々にお礼申し上げます．

参 考 文 献

1) McCarthy, J.: PHENOMENAL DATA MINING: FROM DATA TO PHENOMENA, *ACM SIGKDD Explorations*, Vol.1, No.2, pp.24–29 (2000).
2) Martin, H.C.L., Zhang, N. and Anil, K.J.: Nonlinear Manifold Learning For Data Stream, *Proc. SIAM International Conference for Data Mining*, pp.33–44 (2004).
3) Scholkopf, B. and Smola, A.J.: *Learning with Kernels*, MIT Press (2002).
4) Shawe-Taylor, J. and Cristianini, N.: *Kernel Methods for Pattern Analysis*, Cambridge University Press (2004).
5) Jain, A., Zhang, Z. and Chang, E.Y.: Adaptive non-linear clustering in data streams, *CIKM '06: Proc. 15th ACM International Conference on Information and Knowledge Management*, New York, NY, USA, pp.122–131, ACM (2006).
6) Milenova, B.L., Yarmus, J.S. and Campos, M.M.: SVM in Oracle Database 10g: Removing the Barriers to Widespread Adoption of Support Vector Machines, *VLDB '05: Proc. 31st International Conference on Very Large Data Bases*, VLDB Endowment, pp.1152–1163 (2005).
7) 都築　学，小西　修，新美礼彦：カーネル法による現象データマイニング，電子情報通信学会第 18 回データ工学ワークショップ (2006).
8) 鈴木秀男，水野　誠，住田　潮，佐治　明：CRM のための優良顧客識別手法の特性評価と財務効果，*Department of Social Systems and Management Discussion Paper Series*, No.1123 (2005).
9) Hoi, S.C.H., Jin, R. and Lyu, M.R.: Large-scale text categorization by batch mode active learning, *WWW '06: Proc. 15th International Conference on World Wide Web*, pp.633–642, ACM (2006).
10) Komarek, P. and Moore, A.: Making Logistic Regression A Core Data Mining Tool: A Practical Investigation of Accuracy, Speed, and Simplicity, Technical report, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA (2005).

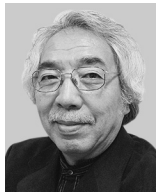
11) Joachims, T.: Text categorization with support vector machines: Learning with many relevant features, *Proc. European Conference on Machine Learning* (*ECML'98*), pp.137–142 (1998).
12) 鹿島久嗣：カーネル法による構造データマイニング，情報処理，Vol.46, No.1, pp.27–33 (2005).
13) 津田宏治：カーネル設計の技術，情報論的学習理論ワークショップ（IBIS2002），pp.1–10 (2002).
14) Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N. and Watkins, C.: Text classification using string kernels, *Journal of Machine Learning Research*, Vol.2, pp.419–444 (2002).
15) Haussler, D.: Convolution Kernels on Discrete Structures, Technical report, UC Santa Cruz (1999).
16) Joachims, T.: SVM-Light Support Vector Machine. http://svmlight.joachims.org/

(平成 20 年 6 月 20 日受付)
(平成 20 年 10 月 7 日採録)

（担当編集委員　森本 康彦）

都築　学（学生会員）

1985 年生．2007 年公立はこだて未来大学システム情報科学部複雑系科学科卒業．現在，同大学大学院システム情報科学研究科システム情報科学専攻博士前期課程に在学中．



小西　修（正会員）

公立はこだて未来大学システム情報科学部教授．1966 年立命館大学理工学部卒業．京都大学工学博士．京都大学工学部，名古屋大学プラズマ研究所，文部省核融合科学研究所，高知大学理学部を経て，2002 年より現職．2008 年より公立大学法人公立はこだて未来大学理事・副学長．主にデータからの学習，創発型情報システムに関する研究に従事．ACM，IEEE-CS，電子情報通信学会，日本データベース学会各会員．