Vol.2012-MPS-91 No.26
Vol.2012-BIO-32 No.26
2012/12/7

ランダマイズドアルゴリズムによる局所線形SVMの並列化

河村 勇太[1,a)]　上原 邦昭[1]

概要：線形 SVM は高速な学習が可能であるが，線形分離不可能なデータには分類精度が低い．一方，カーネル SVM は高い分類精度が得られるが，特徴量の次元が増大することにより学習時間が膨大になる．これらの欠点を解決した局所線形 SVM は，計算量を抑えたまま，既存のカーネル SVM に劣らない分類精度を得られる手法である．本稿では，局所線形 SVM を改良し，さらに学習を高速化させる並列化手法を提案する．並列化手法として，データの分割学習を採用しているが，ノード間のパラメータに差が発生することによる分類精度の低下が問題となる．また，パラメータ差を解消するためにノード間の全通信を行うと，通信コストが膨大になるため，通信回数を少数に絞った並列化を導入する必要がある．本研究では，Cut-And-Stitch のアルゴリズムとランダマイズドアルゴリズムを組み合わせ，通常の学習と殆ど変わらない学習結果が高確率で得られることを実験により示す．

Parallelization of Locally linear Support Vector Machine Using Randomized Algorithm

Abstract: Linear SVMs can be efficiently trained but they suffer from low classification accuracy on non-linearly separable data. Kernel SVMs, on the other hand, can obtain high classification accuracy, but are computationally more expensive to train. Locally Linear SVM is a method that can obtain classification accuracy as good as kernel SVM without compromising training efficiency. In this paper, we further accelerate it by parallelization. However, drop in classification may appear due to parameter difference among processing nodes. All-to-all communication is the best way to solve this problem. However, it requires higher communication cost between nodes. To solve this problem, in this paper, we combine the cut-and-stitch algorithm and a randomized algorithm and show that we can obtain classification accuracy as good as normal Locally Linear SVM with high probability.

1. はじめに

近年の機械学習の特徴として，使用データの大規模化や高次元化が挙げられ，計算量や使用メモリの増加が顕著となってきている．本研究の目的は，線形分離不可能な複雑なデータにも対応可能，かつ高速に学習可能な分類器の並列化手法を検討することである．

分類問題はパターン認識の基本的問題であり，識別器としてニューラルネットワークや SVM 等がよく用いられる．SVM (Support Vector Machine) は，認識性能に優れた学習モデルの 1 つであり，２つのクラスを分類する際に，訓練データとのマージンを最大化する超平面を学習する手法である [1]．SVM は，線形 SVM とカーネル SVM の 2 種類に大別される．線形 SVM は，直線の超平面で分類を行う最もシンプルな手法であり，計算量が少ないという利点があるが，線形分離不可能なデータを分類する際には，大きく分類精度を下げてしまうという性質がある．一方，カーネル SVM は，カーネル関数を使用して非線形の超平面を作成するため，空間的に複雑なデータにも高い分類精度を発揮するが，次元数の増加による計算量の増大が問題となる．これらの問題を解決するため，局所線形 SVM が提案されている [2]．局所線形 SVM は，学習データからアンカー点を作りだし，各アンカー点に対応させた線形 SVM を重ね合わせて，新たな超平面を作成するものであり，計算量を抑えたまま，既存のカーネル SVM に劣らない分類精度が獲得可能となる．

1　神戸大学大学院
Graduate School of System Informatics, Kobe University
a)　k-mura@ai.cs.kobe-u.ac.jp

ⓒ 2012 Information Processing Society of Japan

機械学習では，並列化による高速化がよく行われるが，その方法はデータ並列とタスク並列に大別される．データ並列では，使用するデータを分割して同一の処理が行われる．タスク並列では，順番に依存しないタスクが並列して実行される．局所線形 SVM の学習をタスク並列化しようとすると，タスクの順番に依存性があり，処理の内部で行っている計算の負荷も均一ではないため，並列化が困難である．一方，局所線形 SVM をデータ並列化すれば，各ノードの扱うデータ量が減少するため，さらに高速化すると考えられる．しかし，局所線形 SVM により学習するデータ自体には直接的な依存関係は無いものの，学習に用いるパラメータはデータ毎に更新されるため，それぞれのノードが持つ SVM のパラメータに差が発生してしまい，パラメータの統合によって分類精度が低下するという問題がある．

一方，線形動的システムの並列化に手法として Cut-And-Stitch [3] が提案されている．線形動的システムとは，系列データとそれに対応する隠れ変数を持ち，各隠れ変数は直前の隠れ変数の状態に依存した，1 次元の鎖状構造となる学習モデルである．この依存関係のために，途中の隠れ変数のパラメータを学習するには，最初のデータから順番に学習を行う必要がある．このため，通常の並列処理は適用できない．この問題を解決するため，学習モデルをサブモデルに分割し，サブモデル内の学習が終わると，隣接したサブモデルにパラメータを渡し，擬似的な依存関係を保ちながら，並列処理を可能とする手法が Cut-And-Stitch のアプローチである．

局所線形 SVM の並列化で発生するパラメータの差を解消するためには，全ノードでパラメータの全通信を行うのが最良の方法であるが，通信量が膨大となるために好ましくない．そこで，並列化によって発生するパラメータ差を，サブモデル間で解消するべき依存関係と見なし，Cut-And-Stitch のアイデアを採用した，新たな局所線形 SVM を提案する．具体的には，並列処理を行うノードを，1 個のマスターノードとその他のスレーブノードに分割する．スレーブノードは線形動的システムのように鎖状にリンクを繋ぎ，先頭と最後尾を接続したリング構造にする．各ノードにデータを分割して並列で学習を行う際に，1 つのデータを学習する度に少量の通信を行い，ノード間のパラメータ差を解消する．すなわち，スレーブノードはリンクの張られた両隣のノードと通信を行い，マスターノードはランダムに選択された少数のスレーブノードだけに通信を行う．通信にランダム性を導入することにより，ネットワークの構造による分類精度への悪影響の可能性を平均的に改善することができる．これにより，Cut-And-Stitch のように依存関係を保った並列化が可能となり，ノード当たりの通信量が減少するため，ネットワーク全体の通信量を抑えることができる．最後に，通常の局所線形 SVM と殆ど変わらない分類精度が高確率で得られることを示す．

2. 関連研究

本研究では，局所線形 SVM を改良するために，並列化とランダマイズドアルゴリズムの考え方を使用している．機械学習には，様々な並列化のアプローチが存在するが，タスク並列とデータ並列に大別される．データ並列では，使用するデータを分割して同一の処理が行われる．タスク並列では，順番に依存しないタスクが並列して実行される．一方，ランダマイズドアルゴリズムを使用した機械学習の 1 つとして，ランダムサンプリングが挙げられる．これは，データセットからランダムにデータを取得してサブセットを作成し，そのサブセットで学習することを繰り返す手法である．この手法は，一度に扱うデータ量が少ないためにメモリ効率が良く，処理を数回行えば，データ全体を処理した結果とほぼ同等な結果が得られることが多くなる手法である．以下に，それぞれの手法を用いた関連研究を挙げる．

MapReduce [4] は，Map ステップで細分化したデータを複数のワーカーノードに振り分けて処理を行い，Reduce ステップで処理結果をマスターノードに統合して結果を出力するフレームワークである．Google 社が考案した分散データ処理技術であり，様々な学習手法に適用可能である [5]．MapReduce はデータ並列に分類されるため，本研究の並列局所線形 SVM と同様な処理を行っていることになる．

SVM に並列化を導入したものとして，Cascade SVM [6] がある．この手法では，分割したデータを複数のノードに割り当てて学習を行い，ノード内のデータ中でサポートベクターとなるデータ以外を破棄する．学習が終わると，二つのノードの結果を 1 つに統合して，再びサポートベクターを選択し，それ以外を破棄する．これを繰り返して，最終的なサポートベクターを取得するものである．最終的な学習を行う前に大半のデータを破棄することができるため，通常の学習よりも高速である．Cascade SVM は分割したデータに対する多層分類器を作成しているため，データ並列に分類される．

SVM にランダムサンプリングを使用して，扱うデータ量を削減して実行する研究がある [8]．これは，データセットからランダムにサブセットを作成し，学習によってサポートベクターを得た後，次回のサブセットには，ランダムサンプリングしたデータと前回のサポートベクターを一緒に使用して学習を行う手法である．これは [6] と似た手法であるが，サブセットが自由に作れるため，ノード間の負荷を調整しやすい分，並列化に向いているといえる．

ランダムサンプリングを用いた SVM を，さらに並列化した研究も行われている [9]．この手法では，並列化した各ノードに優先度を設定しており，各ノードでランダムサンプリングしたサブセットによる学習結果を他ノードに全通信し，自身よりも優先度の高いノードから送られた情報を

ⓒ 2012 Information Processing Society of Japan

もとに，自身のパラメータを更新する手法である．サンプリングを行って学習するという処理を並列実行するため，この手法はタスク並列に分類される．この手法に用いられた優先度の概念は，本研究におけるマスターとスレーブの2種ノードの関係性と近い．

また，ランダマイズドアルゴリズムを導入している他の例として，SVM の学習にランダムフーリエ変換を施したカーネル関数を用いて，通常のカーネルより効率よく高次元空間にデータの写像を行い，非線形分類の高速化を行う研究がある [10]．この手法は，フーリエ変換を行ったカーネル関数のパラメータをランダムに設定して，写像後の入力データの特徴ベクトルを作成する手法である．大規模データに対応するカーネル関数は非現実的であるが，この研究の手法を用いれば，カーネル関数自体に変更を加えることにより，実現可能なものに改良できる．

3. 局所線形 SVM

局所線形 SVM とは，通常の線形 SVM を複数個重ね合わせて複雑な超平面を作成し，線形 SVM の速度を活かしながら，線形分離不可能なデータにも対応可能となる SVM である．この手法は，入力したデータの値に応じて超平面が変化することにより，入力データ数と同じ数の超平面が生成される．直線群の包絡線が曲線を描くように，超平面群全体で作られる超平面は曲線を描くことが可能である．

通常の線形 SVM は，重みベクトル $\mathbf{W}$ とバイアス b の２つをパラメータとして持ち，超平面を表現している．一方，局所線形 SVM は複数個のアンカー点を持ち，各アンカー点にそれぞれ 1 つの線形 SVM が対応する．アンカー点とは，訓練データを代表する点であり，クラスタリング後の各クラスタの中心点などが該当する．局所線形 SVM の超平面は，アンカー点が持つ SVM のパラメータを重ね合わせて表現される．その際に，多様体学習 [11] の手法が用いられる．多様体とは，局所的にユークリッド空間と見なせるような図形を意味している．特徴量が膨大なデータに対して，どの特徴量を削除すれば最もデータを表現しやすくなるかを学習する手法が多様体学習である．多様体学習は，次元削減や可視化によく用いられる．

多様体学習では，多様体上のいかなる点 $\mathbf{x}$ も，その点の周囲のアンカー点 $\mathbf{v}$ の線形結合で近似する手法がよく用いられる．

$$\mathbf{x} \approx \sum_{\mathbf{v}} \gamma_{\mathbf{v}}(\mathbf{x})\mathbf{v} \tag{1}$$

γ は係数ベクトルであり，アンカー点 $\mathbf{v}$ と入力ベクトル $\mathbf{x}$ の距離に応じて重み付けされ，$\sum_{\mathbf{v}} \gamma_{\mathbf{v}} = 1$ を満たすものとする．

同様に，低次元の多様体上に定義されたリプシッツ関数 $f(\mathbf{x})$ は，アンカー点 $\mathbf{v}$ の関数値 $f(\mathbf{v})$ の線形結合で近似できるという特性がある．そこで，以下の近似が定義される．

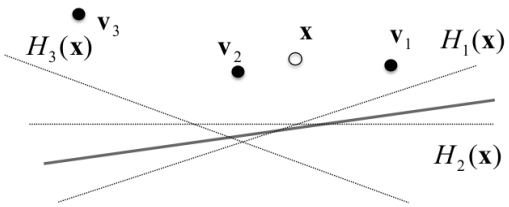

図 1 超平面のイメージ
Fig. 1 Image of hyperplane

$$f(\mathbf{x}) \approx \sum_{\mathbf{v}} \gamma_{\mathbf{v}}(\mathbf{x}) f(\mathbf{v}) \tag{2}$$

この近似式を用いて，局所線形 SVM を導出する．

通常の線形 SVM は以下の式で表される．

$$H(\mathbf{x}) = \mathbf{W}^{\mathbf{T}}\mathbf{x} + b = \sum_{i=1}^{n} w_i x_i + b \tag{3}$$

この $\mathrm{H}(\mathbf{x})$ を超平面とし，パラメータ $\mathbf{W}$ と b を学習する．局所線形 SVM では，この線形 SVM のパラメータ $\mathbf{W}$ と b が，入力ベクトル $\mathbf{x}$ の位置によって変化するパラメータ $\mathbf{W}(\mathbf{x})$, $\mathrm{b}(\mathbf{x})$ に変更される．

$$H(\mathbf{x}) = \mathbf{W}(\mathbf{x})^{\mathbf{T}}\mathbf{x} + b(\mathbf{x}) = \sum_{i=1}^{n} w_i(\mathbf{x}) x_i + b(\mathbf{x}) \tag{4}$$

$\mathbf{W}(\mathbf{x})$, $\mathrm{b}(\mathbf{x})$ は特徴空間 $\mathbf{x}$ においてリプシッツ連続であるため，入力ベクトルを少数のアンカー点 $\mathbf{v}$ によって近似させることができる．よって式 (2) と式 (4) より，局所線形 SVM は以下の式で表される．

$$H(\mathbf{x}) = \sum_{\mathbf{v}} \gamma_{\mathbf{v}}(\mathbf{x}) \left(\sum_{i=1}^{n} w_i(\mathbf{v}) x_i + b(\mathbf{v}) \right) \tag{5}$$

右辺の括弧内は，アンカー点 $\mathbf{v_i}$ に対応する線形 SVM $H_i(\mathbf{x})$ であり，それらを $\mathbf{v}$ と $\mathbf{x}$ の距離に応じた重み $\gamma_{\mathbf{v}}$ によって線形結合している．

図 1 のように、アンカー点 $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ と，それらに対応した線形 SVM $H_1(\mathbf{x}), H_2(\mathbf{x}), H_3(\mathbf{x})$ を持った場合を考える．入力データ $\mathbf{x}$ が与えられると，$\mathbf{x}$ と各アンカー点との距離を測定し，アンカー点 $\mathbf{v}_i$ と $\mathbf{x}$ の距離が近いほど，重み γ_i が大きくなるように設定される．図 1 では，$\mathbf{v}_1$ と $\mathbf{v}_2$ が $\mathbf{x}$ と近いため，γ_1 と γ_2 は大きくなり，γ_3 は小さく設定される．最終的に，設定された γ_i と各アンカー点の持つ線形 SVM $H_i(\mathbf{x})$ を掛け合わせ，全アンカー点の結果を足し合わせれば，新たな線形 SVM が作成できる．これが，入力データ $\mathbf{x}$ に対応する線形 SVM となる．重み γ を使用することにより，入力データと離れたアンカー点は超平面作成から省くことができる．

図 1 では，重みが大きい $H_1(\mathbf{x})$ と $H_2(\mathbf{x})$ が強く影響した線形 SVM(太線) が作成できている．入力データ毎にこの処理を行うため，入力データの数だけ線形 SVM が作成される．大量の線形 SVM によって非線形な領域を表現す

ⓒ 2012 Information Processing Society of Japan

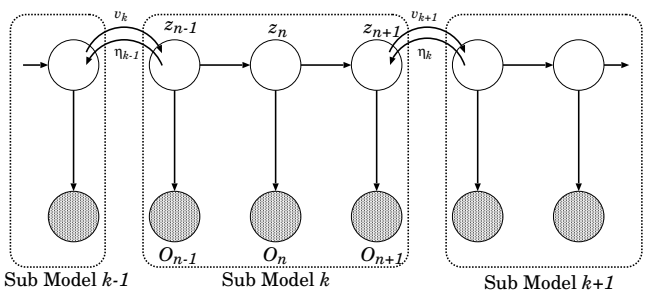

図 2 Cut-And-Stitch のイメージ
Fig. 2 Image of Cut-And-Stitch approach

るため，結果として，複雑な超平面を表現できる．

以上より，局所線形 SVM の学習アルゴリズムは以下のステップに分かれる．

(1) アンカー点の取得
k-means 等のクラスタリング手法を行い，訓練データからアンカー点を取得する．

(2) 線形 SVM のパラメータを学習
訓練データを 1 つずつ読み込んで，SVM のパラメータを更新する．読み込んだ訓練データに近いアンカー点を数個選択し，それらの点が持つパラメータを距離に応じた重みをつけて足し合わせ，新たな線形 SVM を作成する．訓練データが線形 SVM のマージンを下回っていれば，確率勾配降下法を用いて，選択された各アンカー点の持つパラメータを更新する．

局所線形 SVM は，カーネル関数を用いた非線形 SVM と比較して，次元数が増加しないため，短い実行時間で実行可能であるという利点がある．また，複雑な超平面も表現できるため，非線形 SVM と殆ど同じ分類精度を得ることができる．以下では，局所線形 SVM の学習をさらに高速に行うために，並列処理を追加することを検討する．

4. 局所線形 SVM の並列化

局所線形 SVM の学習はタスクの順番に依存性があり，アンカー点ごとに行う計算の負荷も均一ではないため，タスク並列は高速化に不向きである．一方，学習データを各ノードに分割し，各線形 SVM のパラメータを随時通信するデータ並列型のアルゴリズムは，各ノードの扱うデータ量が減少するため，実行時間も減少するという利点がある．しかし，並列処理を行うノードが全対全通信を行う場合，通信量が膨大になるために高速化は見込めない．よって，通信回数を抑制する必要があるが，どのノードと通信を行うかをどのように設定するかが問題となる．この問題を解決するため，Cut-And-Stitch [3] のアイデアを採用する．

Cut-And-Stitch は，データ間に依存関係のある線形動的システムの学習を，依存関係を維持したままデータ分割した並列学習に変換するアプローチである．イメージ図を **図 2** に示す．通常の線形動的システムは，図 2 内のサブモデル k のようになっており，隠れ変数 z_n は直前の観測変数 z_{n-1} の状態に依存し，観測関数 O_n は z_n に依存する．したがって，z_n のパラメータを学習するには，z_1 から順に学習をしなくてはならない．Cut-And-Stitch は，データをサブモデルに分割し，サブモデル内での学習が終わると，依存関係を維持するために，学習に必要なパラメータ v と η を隣接したサブモデルに渡し，擬似的な依存関係の構築を行う．これにより，サブモデルに分断しない状態と殆ど変わらない結果を出力できる並列化手法である．

通常の局所線形 SVM の学習は，入力データ毎に逐次的にパラメータを更新するため，直前のデータまでの学習結果の積み重ねによって精度が向上する．言い換えると，あるデータを学習した時点での分類精度は，直前の学習でのパラメータに依存している．データ並列による並列化を行った場合，この依存関係は断たれてしまい，本来ならば蓄積されているはずの，直前の学習結果が失われることになり，分類精度に影響を及ぼしてしまう．このように，局所線形 SVM の学習は，線形動的システムが持っている，鎖状構造ようなデータ依存関係と考えることができるため，Cut-And-Stitch のアプローチの導入することを考える．具体的には，並列した各ノードを Cut-And-Stitch でのサブモデルに対応させ，分割した学習データを割り当てる．そして，ノード内で学習を行う度に，両隣のノードにのみ通信を行う．また，先頭のノードと最後尾のノード間にも通信を行わせることにより，余分な待機時間のないリング構造に変換する．しかし，リング構造の対極に位置するノード間では，必然的に距離が遠くなるために，パラメータ差が解消されにくく，このままでは分類精度の向上が見込めない．このため，リング構造をさらに改良する必要がある．

4.1 効率的なネットワーク

局所線形 SVM の並列化において，ノード間の通信をリンクとして表現すると，パラメータの広がりはネットワーク上での情報伝播に置き換えることができる．また，並列化によって発生するパラメータ差を解消するには，各ノードが学習したパラメータを効率よくネットワーク全体に伝播させる必要がある．情報伝播の効率性には，ネットワークの構造が大きく作用するため，局所線形 SVM に適した効率的なネットワーク構造を考える必要がある．

感染症流行予測などのシミュレーションの分野では，数理モデルが広く用いられている．その中で，実世界の人間関係などの特徴である，スモールワールドの理論がある [12]．この理論は，あるネットワーク中に少数のランダムなリンクを張れば，ネットワーク全体の直径が極めて短くなるというものである．この理論を局所線形 SVM に採用し，Cut-And-Stitch で作成したリング構造に，ランダムな通信を少量追加して，各ノード間の距離を短くし，ノード間のパラメータ差を解消する手法を検討する．

ⓒ 2012 Information Processing Society of Japan

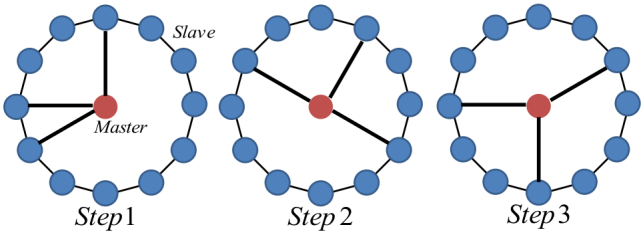

図 3 ランダムリンク作成のイメージ
Fig. 3 Image of making Random Link

4.2 ランダマイズドアルゴリズム

機械学習において，ランダマイズドアルゴリズムは，最悪時という概念を無くし，いかなる場合にも，平均的に良い性能を示すために用いられる．この性質を利用して，リング構造上のノード間でランダムなリンク作成を行い，結果への悪影響の可能性を平均的に改善することを考える．

ランダムリンクを作成するためのノードとして，新たにマスターノードを作成し，リング構造上のノードをスレーブノードと位置づける．マスターノードは，1 つのデータを学習し終えるステップことに，スレーブノードからランダムに数カ所選択し，リンクを構成する．イメージ図を**図 3** に示す．各ノードに割り振られたデータを 1 つ学習する度に，中央のマスターノードから周囲のスレーブノードへリンクをランダムに生成する．リンクが作成されると，各ノードは自身のパラメータをリンク先に送信し，同時にリンク先からパラメータを受け取る．そして，自身の持っていたパラメータと受け取ったパラメータを統合する．これを 1 ステップとし，次のステップでデータを学習する際のパラメータとして，さらに学習を進めていく．

4.3 局所線形 SVM の並列化

本研究で作成する並列化を用いた局所線形 SVM は，Cut-And-Stitch のサブモデルのように，各ノードを鎖状に繋ぎ，先頭と最後尾を繋いでリング構造にしている．このリング構造上のノードをスレーブノードとし，それらとは独立したマスターノードを 1 つ作成する．学習データは全てのノードに分散させ，全てのノードで並列学習を行う．その際に，マスターノードは，スレーブノードへの一定数のリンクをランダムに生成する．ランダムリンクの生成は，各ノードが 1 つのデータを学習する度に行う．これにより，情報のネットワーク伝播性の向上が見込まれる．また，通信の負荷を一定にするため，生成するランダムリンクの本数はあらかじめ設定しておく．

データを 1 つずつ読み込んで学習を行い，ネットワーク内でリンクしているノード間で通信を行うことを繰り返し，各ノード間のパラメータ差を解消しながら学習する．マスターノードは，ランダムリンクを作成することと同時に，自身を基点としてスレーブノードの情報の集積と分散を行うことを役目とする．ノードの増加により，全てのノードを同等に扱う場合よりノード間平均距離は増加してしまうが，マスターノードから多ノードへの平均距離を低下させることができる．最終的なパラメータは，マスターノードの値を用いて，テストデータに適用する．

文献 [2] では，手書き数字データセット（MNIST）を使用して局所線形 SVM の学習を行い，分類精度 98.15%を 81.7 秒で達成している．これを再現しようと試みたが，分類精度 95.40%，実行時間 46.2 秒となった．したがって，これより短い時間でこの分類精度に近づけることが，本研究の中間目標となる．

5. 実験

実験環境の都合上，並列ノード数を 16 に設定して実験を行う．実験には，MNIST と呼ばれる手書き数字データセットを使用して分類を行う．MNIST は 60,000 個の訓練データと 10,000 個のテストデータから構成されており，1 個のデータは 784 次元の実数値を持っている．このデータセットを各ノードに均等に分割をして，並列に学習を行う．アンカー点の数を 100 個に設定し，そのうち入力データに近い 8 個を超平面作成に使用する．アンカー点の数値は文献 [2] で用いられたパラメータであり，MNIST の学習に十分な量である．アンカー点は，通常の k-means における各クラスタの中央値を使用する．

5.1 ランダムリンクの個数の設定

マスターノードからスレーブノードへの適切なリンク数を得るため，リンク数を推移させた比較実験を行う．リンク数が大きいほどパラメータの統合と分散が良好となるが，通信にかかる時間は増加してしまい，リンクされていないスレーブノードの待機時間が増加してしまう．そのため，リンク数は 1~8 までの範囲に絞って推移させ，それぞれで複数回の実験を行う．データ並列によってノードに割り当てられた学習データを学習し終えた状態での分類精度と実行時間を比較する．実行時間の項目は，並列化を用いた SVM の部分のみの実行時間を比較している．結果を**表 1** に示す．分類精度で比較すると，リンク数 1 と 2 の分類精度が少し低く，リンク数 3 以降に大きな差が見られない．実行時間で比較すると，リンク数 7 と 8 が少し実行時間が長く，他のリンク数の場合には大きな差が見られない．よって，3~6 中からランダムリンクの本数を選択することにする．表 1 で最も高い分類精度を得た，リンク数 6 を今後の実験でのランダムリンク数とする．

5.2 データ数の変化による分類精度の推移

5.1 の実験により，ランダムリンクの本数を 6 本と設定して学習を行う．学習による分類精度の算出を数回行い，その平均を**図 4** に示す．最も精度の高かった地点の分類精度は 95.29%であり，目標値にせまる結果となった．

ⓒ 2012 Information Processing Society of Japan

表 1 ランダムリンク数の推移による実行結果の比較
Table 1 Comparison of the performance by shifting the number of random links

リンク数	1	2	3	4	5	6	7	8
実行時間 (millisec)	3610.2	3658.0	3713.0	3728.8	3701.4	3754.4	3902.8	3930.4
分類精度 (%)	94.34	94.72	95.14	95.11	95.08	95.25	95.10	95.21

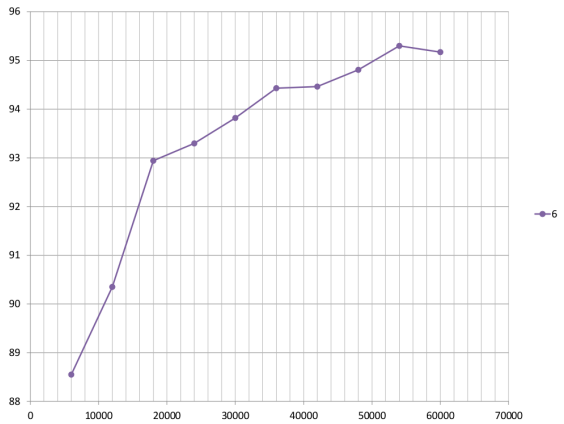

図 4 分類精度の推移
Fig. 4 Transition of classification accuracy

表 2 他手法との比較
Table 2 Comparison of other methods

手法	分類精度 (%)	実行時間 (s)
LibSVM	98.66	17500
線形 SVM	88.00	1.5
局所線形 SVM	98.15	81.7
並列局所線形 SVM	95.25	11.0

5.3 既存手法との比較

実行時間と分類精度の観点から，他の既存手法との比較を行う．結果を**表 2** に示す．LibSVM [13] は，カーネル関数を使用した非線形 SVM であり，既存手法の中でも高い分類精度を誇る手法である．表 2 より，通常の局所線形 SVM よりも高速であり，線形 SVM よりも高い分類精度の学習が可能であることが分かった．本研究の目的は，局所線形 SVM に近い分類精度を得られる分類器を，より高速に学習することであった．しかし，今回作成した並列化局所線形 SVM の精度は，通常の局所線形 SVM より分類精度がかなり劣っている．これは，文献 [2] の局所線形 SVM を元に作成した SVM の分類精度が悪く，それを元に研究を行ったためであると考えられる．

6. 結論

局所線形 SVM に並列化とランダマイズドアルゴリズムを用いて，新たなデータ並列学習の構造を提案した．また，分類精度は他の既存手法に多少劣るものの，短い時間で学習が可能な分類器を作成できた．しかし，文献 [2] の局所線形 SVM と比較して，精度がかなり劣っている．

今後の課題点として，並列化を用いない局所線形 SVM の分類精度の向上が挙げられ，アンカー点の取得方法が一番の問題点であると考える．パラメータのノード間統合の方法も，分類精度に大きく影響を与える部分である．この 2 つの問題点を改善する必要がある．また本研究は，ランダム性を取り入れた学習を行っている．そのため，結果に悪影響を及ぼす可能性の上界を導出する必要がある．よって今後は，Chernoff の定理等を用いて証明を行っていく予定である．

参考文献

[1] 元田浩, 津本周作, 山口高平, 沼尾正行 共著: “データマイニングの基礎”, オーム社 (2006).
[2] L’ubor Ladicky and Philip H.S. Torr: “Locally Linear Support Vector Machines”, *Proc. of the 28th ICML*, pp.985-992 (2011).
[3] Lei Li, Wenjie Fu, Fan Guo, Todd C. Mowry and Christos Faloutsos: “Cut-And-Stitch: Efficient Parallel Learning of Linear Dynamical Systems on SMPs”, *Proc. of the 14th ACM SIGKDD*, pp.471-479 (2008).
[4] Jeffrey Dean and Sanjay Ghemawat: “Mapreduce: Simplified Data Processing on Large Clusters”, *Proc. of the 6th OSDI*, pp.137-150 (2004).
[5] C. Chu, S. Kim, Y. Lin, Y. Yu, Gary Bradski, Andrew Y. Ng, and Kunle Olukotun: “Map-Reduce for Machine Learning on Multicore”, *Proc. of the 20th NIPS*, pp.281-288 (2006).
[6] Hans P. Graf, Eric Cosatto, Leon Bottou, Igor Dourdanovic and Vladimir Vapnik: “Parallel Support Vector Machines: The cascade SVM”, *Proc. of the 19th NIPS*, pp.521-528 (2005).
[7] Michael Mitzenmacher, Eli Upfal 著, 小柴健史, 河内亮周 訳: “確率と計算 -乱択アルゴリズムと確率的解析-”, 共立出版 (2009).
[8] Krishnan S, Chiranjib Bhattacharyya, Ramesh Hariharan and Strand Genomics: “A Randomized Algorithm for Large Scale Support Vector Learning”, *Proc. of the 22nd NIPS*, pp.793-800 (2008).
[9] Yumao Lu and Vwani Roychowdhury: “Parallel Randomized Support Vector Machine”, *Proc. of the 10th PAKDD*, pp.205-214 (2006).
[10] Ali Rahimi and Benjamin Recht: “Random Features for Large-Scale Kernel Machines”, J. C. platt, D. Koller, Y. Singer and S. Roweis, (eds.) *Advances in Neural Information Processing Systems*, Vol 20, pp. 1177-1184, MIT Press (2008).
[11] Lawrence Cayton: “Algorithms for Manifold Learning”, Technical report, University of California (2005).
[12] Barabasi Albert-Laszlo 著, 青木薫 訳: “新ネットワーク思考 -世界のしくみを読み解く”, NHK 出版 (2002).
[13] C. Chang and C. Lin: “LIBSVM: A Library for Support Vector Machines”, Software, http://www.csie.ntu.edu.tw/ cjlin/libsvm (2001).

ⓒ 2012 Information Processing Society of Japan