論文 / 著書情報
Article / Book Information

題目(和文)	頑健かつオンライン学習可能なノンパラメトリック密度推定法
Title(English)	
著者(和文)	中村圭宏
Author(English)	Yoshihiro Nakamura
出典(和文)	学位:博士（工学）, 学位授与機関:東京工業大学, 報告番号:甲第10230号, 授与年月日:2016年3月26日, 学位の種別:課程博士, 審査員:長谷川 修,渡邊 澄夫,樺島 祥介,新田 克己,小野 功
Citation(English)	Degree:, Conferring organization: Tokyo Institute of Technology, Report number:甲第10230号, Conferred date:2016/3/26, Degree Type:Course doctor, Examiner:,,,,
学位種別(和文)	博士論文
Type(English)	Doctoral Thesis

東京工業大学 大学院

学位論文

頑健かつオンライン学習可能なノンパラメトリック密度推定法

総合理工学研究科
知能システム科学専攻
中村 圭宏

指導教員　長谷川 修 准教授

平成28年2月

目 次

図一覧

表一覧

アルゴリズム一覧

記号一覧

$\boldsymbol{\xi}$	入力サンプル．$\boldsymbol{\xi} \in \mathbb{R}^d$.
$\mathcal{N}$	全ノードの集合.
$\mathcal{E}$	全エッジの集合．$\mathcal{E} \subset \mathcal{N} \times \mathcal{N}$.
$\mathcal{P}_i$	ノード i の隣接ノード集合.
$\boldsymbol{w}_i$	ノード i の位置ベクトル．$\boldsymbol{w}_i \in \mathbb{R}^d$.
t_i	ノード i が競合学習において勝者になった回数.
a_e	エッジ e の年齢.
Θ_i	エッジ，ノード作成におけるノード i の閾値.
$\psi_k(t)$	競合学習の勝者ノードの更新における係数決定のための関数. $\psi_1(t) = t^{-1}$，$\psi_2(t) = (100t)^{-1}$
λ	ノード削除ハイパーパラメータ.
age_{max}	エッジ削除ハイパーパラメータ.
ρ	閾値領域補正ハイパーパラメータ.
$\boldsymbol{I}$	単位行列.
$\lvert\mathcal{S}\rvert$	集合 $\mathcal{S}$ の要素数.
$E(\mathcal{G})$	グラフ $\mathcal{G}$ のエッジ集合.

1 序論

1.1 背景

1.1.1 大規模データ

近年，膨大なデータが生成されるようになってきている．Turner ら [1] によると，2013 年では全世界で 4.4ZB ($= 4.4 \times 10^{21}$B) のデータが生成されている．更にその量は指数関数的に増加しており，2020 年には 10 倍の 44ZB にまで拡大すると予測されている．このように大規模なデータが生成されるようになり，これらを解析し活用する事に対して関心，期待が高まっている [2–5].

データの生成量の劇的な増加の要因の 1 つとして，モノのインターネット (Internet of Things: IoT) が挙げられる [1, 6]．IoT とは，これまでインターネットに接続していなかった様々なモノがインターネットに接続することである．モノにはセンサーが取り付けられ，データを記録し，インターネットを通じて送受信することで，新たな機能を発揮する．スマートフォンや家電製品，自動車など日常的に利用するモノ、更にはウェアラブルデバイスによって人体にもセンサーが取り付けられ，継続的にデータを生成し続け，インターネットを

通じでデータを送受信するようになってきている．同様のコンセプトとしては Ubiquitous Computing [7] や Cyber-Physical Systems [8] がある．これらは近年のネットワーク技術の普及やセンサーの技術発展と低価格化によって現実的になってきている．

IoT は様々なアプリケーションが考えられる [6]．例えば，複数のウェアラブルセンサーから生成される生体データから，ユーザーの現在の行動や，健康状態，感情などの推論をリアルタイムで行い，それに応じた提案を行うシステムや，車に取り付けられた外界センサーの情報から車がどのように進むか予測し，事故を起こさないようにドライバーを補助するシステムなどが考えられる．

IoT によって継続的にデータを生成し続けるモノの数が増加し続けており，このため生成されるデータの量が指数関数的に増加しているのである．Turner ら [1] の調査によると，インターネットに接続する可能性のあるモノの数は現在 2,000 億個を超えており，このうちの 7% にあたる 140 億個が既にインターネットに接続している．2020 年までにこのようなディバイスの数は 320 億個にまで増加すると予測している．

継続的に生成され続けるデータはストリームデータと呼ばれ研究が行われている [9, 10]．継続的に生成されるということは，潜在的に無限のデータが生成されることを意味している．したがって，全てのデータを保持することは不可能であり，データを取得したら定数時間で処理し，破棄する必要がある．このため，多くの手法はデータの要約するなど簡単な記述統計量を用いている．また，多くの場合，データの分布は非定常で時間とともに変化することを想定している．

1.1.2　データ駆動な知的情報処理

近年の知的情報処理・人工知能の分野において，機械学習というデータ駆動なアプローチが成功を収めている．エキスパートシステムに代表される従来の人工知能はルールベースなアプローチであり，人間が設計したルールを用いて推論などの知的な情報処理を行う．このアプローチでは人間が記述すべきルールが膨大になり，また，実世界の問題にはルールとして記述するのが難しい問題が多くあった．これに対して，機械学習では，データを用いて知的な情報処理を行うシステムを構築する．このアプローチでは，データさえ集めれれば，人間がルールを設計する必要はなく，ルールとして記述するのが難しい問題にも対応できる．

先に述べたとおり，現在では膨大なデータ生成・蓄積されるようになり，且つそれらを処理するだけの計算機の性能向上もあり，より人間に依らず，よりデータ駆動な機械学習のアプローチが成功を収めつつある．近年のニューラルネットワークの成功が，その 1 つである．ニューラルネットワークは任意の関数が近似できるが [11]，モデルが複雑なため学習が困難であり，十分な精度が出ないという問題があった．しかし，大規模な学習データを用いることが可能になったことを一因として，この問題を解決し，様々な分野で高い性能を示している [12–14].

統計的機械学習は確率モデルを用いて，知的な情報処理を設計する．例えば，機械学習の代表的なタスクであるクラス分類問題は，入力データ $\boldsymbol{x}$ が属するクラス $c \in \{c_i\}_{i=1}^{k}$ を推定する問題であるが，これは $\boldsymbol{x}$ に対する各クラス c_i の条

件付き確率 $p(c_i|\boldsymbol{x})$ に対して

$$c = \underset{c_i}{\text{argmax}} \ p(c_i|\boldsymbol{x})$$

と定式化できる．また，説明変数 $\boldsymbol{x}$ から被説明変数 $\boldsymbol{y}$ を予測する回帰問題は条件付き確率 $p(\boldsymbol{y}|\boldsymbol{x})$ を用いて

$$\boldsymbol{y}(\boldsymbol{x}) = E[\boldsymbol{y}|\boldsymbol{x}] = \int \boldsymbol{y} p(\boldsymbol{y}|\boldsymbol{x}) dy \tag{1.1}$$

と定式化できる．そして，確率分布をモデル化し推定することで，問題を解決するシステムを構築する．

確率モデルを設計するには高度な専門知識が必要であり，専門家によって設計される．既存のパラメトリックな分布では複雑さや自由度が足りないため，実際の問題では有効でない場合が多い．そこで，隠れ変数を導入し確率変数を階層構造にすることで，複雑さを表現する．しかし，モデル自体が複雑になり，設計が難しくなる．階層構造を用いることのもう 1 つのメリットは事前知識を導入することができる点である．データに関するドメイン知識を用いて設計することで，より適切な確率モデルを構築できる．しかし，逆に言うと，事前知識がない場合，適切な設計が難しくなる．

今後，データがより膨大に生成されるようになれば，よりデータ駆動な知的処理が重要となってくると考えられる．十分な学習データがあれば，複雑で自由度が高いモデルも学習でき，人間が事前知識や経験則を用いてモデルを設計するよりも高い性能を出すことができるためである．また，先に述べたとおり，IoT の拡大によりデータの多様性は益々増しており，現状でさえも，全てのデータのにおける解析されているデータの割合は 1% 以下であり，解析者の不足が

深刻である．[1]．このため，全てのデータを人間が解析するのではなく，データ駆動な知的処理によって解析を自動化することが必要になると考えられる．

1.2　目的

前章で述べた背景を踏まえ，本研究では IoT などデータストリームから生成される大規模データ活用するための知的情報処理の基盤技術となる確率密度推定法を提案する．これには次の 3 つの特徴を有する必要があると考える:

- 高速なオンライン学習
- ノンパラメトリック
- 高いロバスト性

オンライン学習手法とは，データを逐次的に学習しモデルを徐々に更新していく手法のことである．従って，データが随時増加するような場合にも，追加的に学習することができる．これの対となる学習手法としてバッチ学習手法がある．これは，学習に用いるデータ全てに対して同時に学習を行う手法のことである．一般に，オンライン学習手法よりもバッチ学習手法の方が安定して精度の良いモデルを構築することが可能である．しかし，データが増加した場合，それらのデータを追加で学習することができず，学習を最初からやり直さなければならない．

膨大な量のデータが生成されるようになった一因は IoT などのデータストリームであり，問題はデータの生成される速度，単位時間あたりに生成されるデータ量，である．データが高速に大量に生成され続けているため，結果的にデータの総量が莫大になるのである．バッチ学習では継続的にリアルタイムに

大量に生成されるデータには対応できない．なぜなら，学習が終わる前に新たなデータが生成されてしまい，またデータ量が増えるに従って計算量が大幅に増加してしまうため，学習が間に合わないからである．従って，生成されるデータを逐次的に学習できる高速なオンライン学習手法である必要がある．

ノンパラメトリックな手法とは，データの分布の形状に仮定を置かない手法のことである．これに対して，パラメトリックな手法は，データが何かしらの既知のパラメトリックな分布から発生したと仮定し，そのパラメータを推定することで密度関数を推定する．従って，実際のデータの分布が仮定した分布と異なっていた場合，大幅な推定精度の低下を招いてしまう場合がある．未知の分布や既知のパラメトリックな分布としてモデル化できない分布に対して，パラメトリック密度推定法を用いて密度関数を推定することは困難である．

IoT の拡大によってデータの多様性が増しており，個々のケースに対して専門家がモデリングすることは難しくなってきている．また，データマイニング・知識発見を目的とする場合，多様な種類のデータを用いると共に，いかようなデータが取得できるかを予め予測することが難しい場合が多い．例えば，ウェアラブルセンサーから取得した生体情報などの個体差が大きいデータやパーソナライズが目的で個人に特化する必要がある場合や，新しいセンサーや新しいモノから取得されたデータでまだ専門家によって解析されモデル化されていないデータなどは予め分布を仮定することが難しい．従って，このようなデータには，パラメトリック密度推定法では対応できない．パラメトリック密度推定法では予めパラメトリックな分布を仮定する必要があるためである．これに対して，ノンパラメトリック密度推定法は，分布を仮定せず，データのみから密度関数を推定するため，このような問題は生じない．

ロバスト性とは，ノイズの影響を受けにくい性質のことである．データは実環境から取り入れられたものであり，ノイズを多く含むと考えられる．ノイズを多く含むデータを用いて学習した場合，ロバスト性のない手法では，ノイズに対しても適合してしまい過学習を起こし汎化性能の低下に繋がる．ノイズの分布を仮定することでノイズに対処する事も考えられるが，仮定したノイズの分布が実際のノイズの分布と異なっていた場合，著しい性能の低下を招いてしまう．また，環境が変化した場合や，車や人などの動くモノに設置されたセンサーや日常的なモノに設置されたセンサーなど，様々な環境下でセンシングすることが求められる場合も，この性能低下が発生してしまう．人手でノイズを除去することもできるが，大規模に高速に生成されるデータにおいては非常に困難である．したがって，ノイズを含むデータからでも学習できる高いロバスト性を有する手法が必要である．

ロバスト性に関しては様々な分野で研究されており，各分野でロバスト性の定義が微妙に異なっている [15–18]．本研究においては，ロバスト性を，ノイズを含まない理想的な状態ではないデータを学習した場合でも，理想的なデータの場合と同様の学習結果が得られる性質と定義する．ロバスト性の定義に関しては 第 2.2 章 でより詳細に述べている．

上記の 3 つの特徴全てを有する既存手法は存在しない．代表的なノンパラメトリック密度推定法はカーネル密度推定法 (Kernel Density Estimation: KDE) [19] であるが，多くの手法はバッチ学習手法である [20–25]．オンラインクラスタリング手法を応用するなどして，カーネル密度推定法をオンライン化した手法も考案されているが [26–28]，これらはロバスト性と学習速度が十分とは言えない．カーネル法とロバスト推定を用いてロバストなカーネル密度推定法を実現している手法 [18] もあるが，この手法はバッチ学習手法であり，大規模データには

対応できない．

提案手法は，競合学習型ニューラルネットワークの一種である自己増殖型ニューラルネットワーク (Self-Organizing Incremental Neural Network: SOINN) [29, 30] とカーネル密度推定法を拡張することで，上記の 3 つの性質を実現する．クラスタリング手法である SOINN が形成するネットワークが密度に関する情報を保持していると新たに意味付け，それを用いて密度関数の推定を行う．

1.3　構成

本論文は 6 章から構成されている．各章の主な内容について以下に示す．

第 1 章

序論として本研究の背景と目的を述べる．

第 2 章

代表的なノンパラメトリック確率密度推定法であるカーネル密度推定法について詳細を述べる．カーネル密度推定法は提案手法の基礎となる手法の 1 つである．また，オンライン学習，ロバスト性に関してカーネル密度推定法を拡張した既存手法についても述べる．

第 3 章

本研究の基礎となる手法である自己増殖型ニューラルネットワークについて，その学習原理と，確率密度推定に応用できる統計的な性質に関して述べる．学習原理である Competitive Hebbian Learning とベクトル量子化によって形成されたネットワークが分布をどのように表しているかについて述べる．

第 4 章

第 3 章で述べた SOINN の統計的な性質を応用したノンパラメトリック密度推定法について詳細を述べる．提案手法の推定密度関数および学習アルゴリズムについて示し，評価実験により既存手法と比較し，考察を行う．

第 5 章

第 4 章の結果から，更に学習アルゴリズムの改良による性能向上に関して述べる．SOINN の学習アルゴリズムの問題点を指摘し，その部分をいかに改良したかについて示し，評価実験により既存手法および改良前と比較し，考察を行う．

第 6 章

本論文をまとめ，課題および今後の展望について述べている．

2 カーネル密度推定法

カーネル密度推定法 (kernel density estimation：KDE) [19] は代表的なノンパラメトリック密度推定法である．学習データ $\{\boldsymbol{x}_i | \boldsymbol{x}_i \in \mathbb{R}^d,\ i = 1, 2, \cdots, N\}$ に対する KDE による推定密度関数 $\hat{p}(\boldsymbol{x})$ は，

$$\hat{p}(\boldsymbol{x}) = \frac{1}{N} \sum_{i=1}^{N} K_{\boldsymbol{H}}(\boldsymbol{x}, \boldsymbol{x}_i) \tag{2.1}$$

と表せる．ここで K はカーネル関数と呼ばれ，次のガウスカーネルがよく用いられる:

$$K_{\boldsymbol{H}}(\boldsymbol{x}, \boldsymbol{\mu}) = \frac{1}{\sqrt{(2\pi)^d |\boldsymbol{H}|}} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu})^T \boldsymbol{H}^{-1} (\boldsymbol{x} - \boldsymbol{\mu})\right). \tag{2.2}$$

ここで $\boldsymbol{H}$ はバンド幅と呼ばれるパラメータで半正定値対称行列である．KDE では，各学習データの位置に配置したカーネル関数の線形和としてデータ全体の確率密度関数を推定する．

バンド幅行列 $\boldsymbol{H}$ は性能に大きく影響を与えるため，この最適化に関して多く

の研究が行われている [20–25, 31]．1 変量のバンド幅は良く研究されているが，多変量の場合は，低次元の場合やバンド幅が対角行列や単位行列の定数倍の場合のみを扱う事が多い [31]．[25, 31] では，尤度最大化基準と Leave-one-out 交叉検証法 (Leave-one-out cross-validation: LOOCV) を用いて，全要素が非ゼロの行列であるバンド幅を最適化している．KDE では 式 (2.1) のようにモデルの定義に全学習サンプルを用いるため，尤度が計算できないが，

$$\hat{p}_{\boldsymbol{H}}(\boldsymbol{x}_i) = \frac{1}{N-1}\sum_{j\neq i} K_{\boldsymbol{H}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

と，あるサンプル $\boldsymbol{x}_i$ 以外のサンプルを用いてモデルを近似すれば，

$$L(\boldsymbol{X}|\boldsymbol{H}) = \prod_{i=1}^{N} \hat{p}_{\boldsymbol{H}}(\boldsymbol{x}_i)$$

と尤度を近似できる．そして，$\nabla_{\boldsymbol{H}} log L(\boldsymbol{X}|\boldsymbol{H}) = 0$ から fixed-point rule

$$\boldsymbol{H}_{n+1} = \frac{1}{N(N-1)}\sum_i \frac{1}{\hat{p}_{\boldsymbol{H}_n}(\boldsymbol{x}_i)}\sum_{j\neq i}(\boldsymbol{x}_i - \boldsymbol{x}_j)(\boldsymbol{x}_i - \boldsymbol{x}_j)^T K_{\boldsymbol{H}_n}(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

を導出し，これを反復させることで，バンド幅を最適化する．

上記は，全てのサンプルに配置したカーネルに関して同一のバンド幅を用いているが，密度の高い領域では小さなバンド幅，密度が低い領域では大きなバンド幅を用いた方が適切であると考えられる．そこで，各学習サンプル $\boldsymbol{x}_i$ に対応するカーネルに対して異なるバンド幅 $\boldsymbol{H}_i$ を定義して

$$\hat{p}(\boldsymbol{x}) = \frac{1}{N}\sum_{i=1}^{N} K_{\boldsymbol{H}_i}(\boldsymbol{x}, \boldsymbol{x}_i)$$

という推定密度関数を考える手法も考案されている [32–34]．この場合，単一の

バンド幅を用いる場合と比べて，パラメータの数が大きく増加しており，パラメータの決定がより困難である．[34] では，データが多様体に従っていると仮定することでバンド幅を計算している．$\boldsymbol{x}_i$ の影響の大きさを規定する近傍カーネル $\mathcal{K}(\boldsymbol{x};\boldsymbol{x}_i)$ を用いて，ソフト近傍というものを定義し，

$$\boldsymbol{H}_{\mathcal{K}_i} = \frac{\sum_{j\neq i}\mathcal{K}(\boldsymbol{x}_j;\boldsymbol{x}_i)(\boldsymbol{x}_j-\boldsymbol{x}_i)(\boldsymbol{x}_j-\boldsymbol{x}_i)^T}{\sum_{j\neq i}\mathcal{K}(\boldsymbol{x}_j;\boldsymbol{x}_i)}$$

と，バンド幅 $\boldsymbol{H}_i$ を計算している．

2.1　オンライン学習

KDE では学習サンプル全てに対してカーネルを配置するため，サンプル数の増加に対して，時間計算量，空間計算量共に線形に増加してしまう．しかし，精度を出すためにはサンプル数が必要であり，特に，次元が増加すると必要なサンプル数は指数関数的に増加する [20]．計算量と精度がトレードオフの関係にある．計算量を抑えるために，サンプルの分布の複雑さに応じて適切にカーネルを配置したり，カーネルの数を設定したりする手法がいくつか提案されている [35–37]．

計算量を抑えるために，オンライン化する事も考えられる．オンライン化の方法としては，確率的勾配法 (Stochastic Gradient Descent: SGD) [38,39] が有名である．しかし，KDE に SGD を適応した手法は提案されていない．KDE はサンプルが増えるたびにモデルが変化するため，オンラインに最適化を行うのが困難であるためではないかと推測される．

オンラインクラスタリングなどを用いることで KDE をオンライン化する手法は考案されている [26–28]．oKDE [28] では，全てのサンプルを保持するので

はなく，階層型クラスタリングによって圧縮されたモデルのみを保持し，それをオンラインに更新することで，オンライン化している．新しいサンプルを得ると，そのサンプルを新しいコンポーネントとして加え，全コンポーネントのあるサブセットが，1 つのコンポーネントとして表せる場合，それらのコンポーネントを合成することでモデルを圧縮する．このように圧縮することで，モデルの複雑さ及び計算量を抑えている．

2.2　ロバスト性

式 (2.1) が示すように，KDE では入力サンプル全てに対してカーネルを配置し，その線形和で密度関数を表現する．入力サンプルにノイズを含む場合，ノイズにもカーネルを配置するため，ノイズに対しても適合してしまい過学習を引き起こす一因となってしまう．ノイズに対処する手法としては，ロバスト性を有する KDE として RKDE [18] が提案されている．

ロバスト性に関しては様々な分野で研究されており，各分野でロバスト性の定義が微妙に異なっている [15–18]．本研究においては，ロバスト性を，外れ値や欠損値などのノイズを含まない理想的な状態ではないデータを学習した場合でも，理想的なデータの場合と同様の学習結果が得られる性質と定義する．

ノイズは 2 種類に大別されると考えられる：

(1) 外れ値や異常値，拡散または出現頻度の低い分布から発生したサンプルなどの，目的の分布とは無関係であり，環境から発生したと考えられるノイズ，

(2) クラス内変動，分散，または揺らぎなどの，現象に内在し確率分布として捉えるべきノイズ．

密度推定におけるロバスト性とは（1）のノイズを除去できる性質のことである [18]．つまり，目的の分布 p_{target} とノイズの分布 p_{noise} をある割合 α で混合した分布 p からのサンプル集合

$$\boldsymbol{X} \sim p = (1-\alpha)\, p_{target} + \alpha\, p_{noise},$$

を学習データとして与えられたときに，$\boldsymbol{X}$ から p_{target} を推定できる性質のことである．

RKDE では KDE をカーネル法として捉え，ロバストな推定を行うことでロバスト性を実現している．KDE の推定密度関数 式 (2.1) における $K_{\boldsymbol{H}}$ に対して

$$\Phi(\boldsymbol{x}) \triangleq K_{\boldsymbol{H}}(\cdot, \boldsymbol{x}_i) \tag{2.3}$$

とし，任意の $\boldsymbol{x}$ に対する $\Phi(\boldsymbol{x})$ を要素とする再生核ヒルベルト空間 $\mathcal{H}$ を考える．このとき，KDE の推定密度関数は

$$\begin{aligned} \hat{p}(\cdot) &= \frac{1}{N}\sum_{i=1}^{N} K_{\boldsymbol{H}}(\cdot, \boldsymbol{x}_i) \\ &= \frac{1}{N}\sum_{i=1}^{N} \Phi(\boldsymbol{x}_i) \end{aligned}$$

と $\Phi(\boldsymbol{x}_i)$ の平均として表現できる．したがって，次の最適化問題の解である

$$\hat{p} = \underset{g \in \mathcal{H}}{\operatorname{argmin}} \sum_{i=1}^{N} ||\Phi(\boldsymbol{x}) - g||_{\mathcal{H}}^2. \tag{2.4}$$

RKDE では，式 (2.4) に更にロバストな損失関数 $\rho(x)$ を加え

$$\hat{p} = \underset{g \in \mathcal{H}}{\operatorname{argmin}} \sum_{i=1}^{N} \rho\left(||\Phi(\boldsymbol{x}) - g||_{\mathcal{H}}\right)$$

とすることで，ロバストな KDE を実現している．

このように，ノンパラメトリック密度推定に関して，オンライン学習，ロバスト性のいずれかを考慮した手法は存在する．しかし，これら全ての性質を有する手法はない．

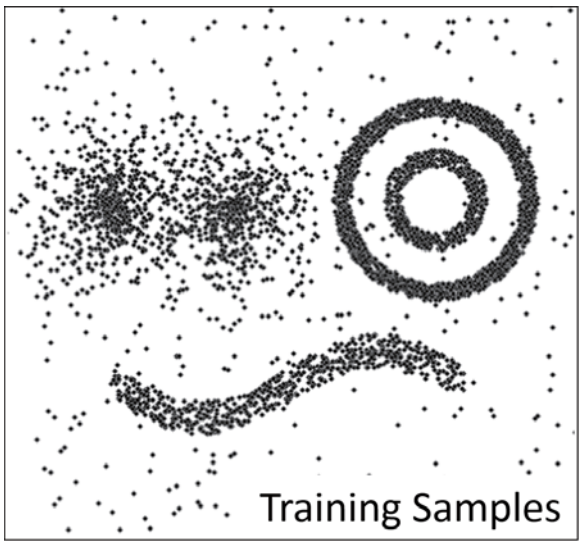
3 自己増殖型ニューラルネットワーク

3.1 概要

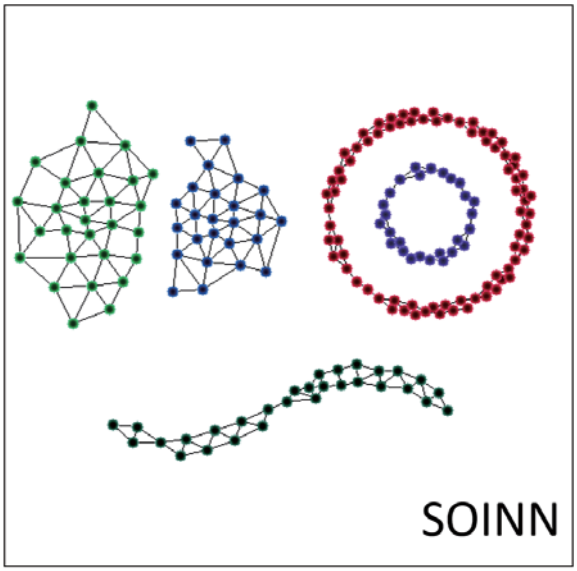
自己増殖型ニューラルネットワーク (Self-Organizing Incremental Neural Network: SOINN) [29] は Growing Neural Gas (GNG) [40] を拡張したプロトタイプベースの教師なし学習手法である．SOINN はオンライン学習手法であり，サンプルを逐次的に学習する．複雑で非定常な分布からのサンプルも，ノイズを除去しつつ，入力サンプル数より大幅に少ないノード数のネットワーク構造として学習できる．ノード数やノードの初期位置を事前に設定する必要がないため追加学習に適しており，既に学習した学習結果に影響することなく，新しく追加されたクラスを学習することができる．

SOINN ではネットワーク構造を用いて入力サンプル集合を学習する．SOINN による学習の例を 図 3.1 に示す．オンラインに入力されるサンプルに対して，ネットワーク上のノードが競合学習を行う．それに伴って，ノードの挿入・削除・位置の更新およびエッジの挿入・削除が必要に応じて行われ，サンプル集合の分布を近似するようにネットワーク構造が自己組織的に更新されていく．

SOINN 改良が重ねられ，複数のバージョンが存在する．オリジナルの SOINN



(a)



(b)

図 3.1: SOINN による学習の例 : (a) ノイズを含む入力サンプル集合 (b) SOINN による学習結果

[29] は，事前にネットワークの形状を決定する必要がなく，非定常な入力を学習することができ，複雑な分布を持つクラスタを，適切なクラスタ数及び位相構造の出力を実現した．更に，ノイズ耐性も実現した．しかし，ネットワークの構造が 2 層構造であり，1 層目は追加学習が可能であるが 2 層目は追加学習が可能ではなく，また，2 層目の学習の開始するタイミングを決定することも難しい．更に，クラスタリングにおいて，分布に高密度の重なりがあるクラスは分離できないという問題もあった．これらの問題を解決し，ネットワークを 1 層構造に簡略化したのが Enhanced-SOINN (ESOINN) [41] である．更に，この ESOINN のハイパーパラメータの数を減らし，学習を容易にした Adjusted SOINN が [30] で提案されている．

Adjusted SOINN は 1 層構造であり，ハイパーパラメータも少なく扱いやすく，近年の SOINN の応用研究で用いられることが多い [42–46]．本研究においても，この理由から Adjusted SOINN を用い，SOINN と書いた場合，Adjusted SOINN を指すものとする．

SOINN のアルゴリズムをアルゴリズム 1 に示す．

アルゴリズム 1: SOINN

if 初回の学習である **then**

　$\mathcal{N} \leftarrow \{c_1, c_2\}$

　　▷ 学習データからランダムに選択したサンプルを位置ベクトルとして持つ 2 つのノード c_1, c_2 で全ノードの集合を初期化

　$\mathcal{E} \leftarrow \phi$

end if

while 入力パターン $\boldsymbol{\xi} \in \mathbb{R}^d$ が存在する **do**

　$s_1 \leftarrow \arg\min_{c \in \mathcal{N}} ||\boldsymbol{\xi} - \boldsymbol{w}_c||$

　　▷ $\boldsymbol{\xi}$ に対する第 1 勝者ノード s_1 を探索

　$s_2 \leftarrow \arg\min_{c \in \mathcal{N} \setminus \{s_1\}} ||\boldsymbol{\xi} - \boldsymbol{w}_c||$

　　▷ $\boldsymbol{\xi}$ に対する第 2 勝者ノード s_2 を探索

　次式によって類似度閾値 $\Theta_{s_1}, \Theta_{s_2}$ を計算:

$$\Theta_i = \begin{cases} \max\limits_{p \in \mathcal{P}_i} ||\boldsymbol{w}_p - \boldsymbol{w}_i|| & (\mathcal{P}_i \neq \phi) \\ \min\limits_{p \in \mathcal{N} \setminus \{i\}} ||\boldsymbol{w}_p - \boldsymbol{w}_i|| & (otherwise). \end{cases} \tag{3.1}$$

　if $||\boldsymbol{\xi} - \boldsymbol{w}_{s_1}|| > \Theta_{s_1}$ or $||\boldsymbol{\xi} - \boldsymbol{w}_{s_2}|| > \Theta_{s_2}$ **then**

　　$\mathcal{N} \leftarrow \mathcal{N} \cup \{\boldsymbol{\xi}\}$　　▷ $\boldsymbol{\xi}$ を新規ノードとして追加

　else

　　if $(s_1, s_2) \notin \mathcal{E}$ **then**　　▷ s_1 と s_2 の間にエッジがない

　　$\mathcal{E} \leftarrow \mathcal{E} \cup \{(s_1, s_2)\}$　　▷ エッジ (s_1, s_2) を追加

　　end if

15: $a_{(s_1,s_2)} \leftarrow 0$ ▷ エッジ (s_1, s_2) の年齢を 0 にリセット

16: $a_{(s_1,i)} \leftarrow a_{(s_1,i)} + 1 \ \ (\forall i \in \mathcal{P}_{s_1})$
▷ s_1 につながる全エッジの年齢をインクリメント

17: $t_{s_1} \leftarrow t_{s_1} + 1$ ▷ s_1 の勝者として選択された回数 t_{s_i} をインクリメント

18: $\boldsymbol{w}_{s_1} \leftarrow \boldsymbol{w}_{s_1} + \psi_1(t_{s_1})(\boldsymbol{\xi} - \boldsymbol{w}_{s_1})$ ▷ s_1 の位置ベクトルを更新

19: $\boldsymbol{w}_i \leftarrow \boldsymbol{w}_i + \psi_2(t_i)(\boldsymbol{\xi} - \boldsymbol{w}_i) \ \ (\forall i \in \mathcal{P}_{s_1})$
▷ s_1 の全隣接ノードの位置ベクトルを更新

20: エッジ $\mathcal{E}_{old} = \{e \mid e \in \mathcal{E},\ a_e > age_{max}\}$ を削除

21: ノード $\{i \mid \exists j \ (i,j) \in \mathcal{E}_{old}, |\mathcal{P}_i| = 0\}$ を削除
▷ 20 行目 において削除されたエッジに接続していたノードのうち，接続エッジがなくなったノードを削除.

22: **end if**

23: **if** 入力パターン数が λ の倍数 **then**

24: $|\mathcal{P}_i| \leq 1$ であるノード i をすべて削除

25: **end if**

26: **end while**

3.2　ネットワークの統計的な意味

SOINN は応用研究においてクラスタリング手法として使われてきた．この文脈において，SOINN のネットワークはノードが同一クラスタに属するかどうかの基準であった．しかし，本研究においては，SOINN のネットワークはデータの分布に関する情報を保持しているとし，この情報を利用することで確率密度推定を行う．

SOINN のネットワークが表している分布の情報に関して，Competitive Hebbian Learning によるプロトタイプ間へのネットワーク生成とベクトル量子化によるデータのプロトタイプ生成とに分けて説明する．

3.2.1　Competitive Hebbian Learning

SOINN ではネットワーク形成の基準として Competitive Hebbian Learning を用いている．Competitive Hebbian Learning は， Martinetz ら [47] によって提唱された学習手法で，競合学習に Hebbian rule を取り入れたものである．

Hebbian rule とは人間の脳内で行われていると言われている学習規則である [48].「細胞 i の軸索が細胞 j を発火させるのに十分近くにあり，繰り返しあるいは絶え間なくその発火に参加するとき，いくつかの成長過程あるいは代謝変化が一方あるいは両方の細胞に起こり，細胞 j を発火させる細胞の 1 つとして細胞 i の効率が増加する」というものである．この接続機構の数学的な表現にはいくつか存在する [49–51]．最も単純な定式化は

$$\Delta \boldsymbol{C}_{ij} \propto y_i \cdot y_j \tag{3.2}$$

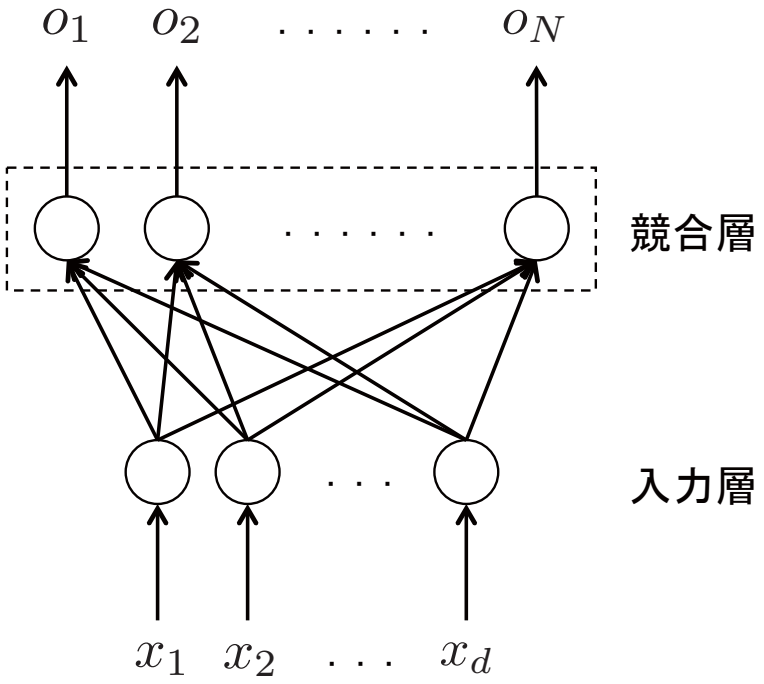


図 3.2: 競合学習モデル

である [52]．ここで $\boldsymbol{C}_{ij}$ はニューロン i, j 間のシナプスの接続強度，y_i, y_j はニューロン i, j の活性，シナプス前活性とシナプス後活性を表している．

競合学習とは，ニューラルネットワークによる教師なし学習手法であり，入力データに対して，複数のユニットが競合し，その勝者のみが学習を行う (winner-take-all)．図 3.2 に競合学習を図示する．競合層のユニット i は重み $\boldsymbol{w}_i \in \mathbb{R}^d$ を保持しており，それを用いて与えられた入力 $\boldsymbol{x} \in \mathbb{R}^d$ に対する活性度を計算する．全てのユニットの中で最も活性度が高かったユニット（勝者ユニット）s_1 のみが $\boldsymbol{x}$ を学習し，s_1 の $\boldsymbol{x}$ に対する活性度がより高くなるように $\boldsymbol{w}_{s_1}$ を更新する．多くの場合，活性度はユークリッド距離によって計算され，この場合の

$\boldsymbol{w}_i$ の更新量 $\Delta\boldsymbol{w}_i$ は

$$\Delta\boldsymbol{w}_i \propto \begin{cases} \boldsymbol{x} - \boldsymbol{w}_i & if \quad \forall k \quad y_i \geq y_k \\ 0 & otherwise. \end{cases} \tag{3.3}$$

となる．ここで，y_i は $\boldsymbol{x}$ に対するユニット i の活性度である．

Competitive Hebbian Learning では Hebbian rule を競合学習の方法論に則り行う．式 (3.2) の Hebbian rule をこの競合学習の文脈で行うと次のようになる:

$$\Delta\boldsymbol{C}_{ij} \propto \begin{cases} y_i \cdot y_j & if \quad \forall k, l \quad y_i \cdot y_j \geq y_k \cdot y_l \\ 0 & otherwise. \end{cases} \tag{3.4}$$

ユニット i に割り当てられている重みベクトルを $\boldsymbol{w}_i \in \mathbb{R}^D$ とする．この $\boldsymbol{w}_i$ はユニット i の受容野の中心に位置し，入力 $\boldsymbol{\xi} \in \mathbb{R}^D$ が $\boldsymbol{w}_i$ に近いほど，ユニット i の活性度 y_i が大きくなる．これは放射基底関数 (Radial basis function: RBF) :

$$y_i = R(||\boldsymbol{\xi} - \boldsymbol{w}_i||)$$

で表現できる．ここで $R(\cdot)$ はガウス関数のような，常に正の値を取り連続で単調減少する関数である．よって，式 (3.4) で示した競合学習において勝利するのは，重みベクトル $\boldsymbol{w}$ が入力 $\boldsymbol{\xi}$ に最も近かった 2 ユニットのペアである．したがって，ネットワークの構成手順は アルゴリズム 2 のようになる．

アルゴリズム 2: Competitive Hebbian Learning

1: 全ユニット i, j 間の接続を未接続とする
2: **while** 入力パターン $\boldsymbol{\xi} \in \mathbb{R}^d$ が存在する **do**
3: 　　$s_1 \leftarrow \arg\min_{c \in \mathcal{N}} ||\boldsymbol{\xi} - \boldsymbol{w}_c||$
4: 　　$s_2 \leftarrow \arg\min_{c \in \mathcal{N} \setminus \{s_1\}} ||\boldsymbol{\xi} - \boldsymbol{w}_c||$
5: 　　ユニット s_1 と s_2 を接続する
6: **end while**

この Competitive Hebbian Learning によって形成されるネットワークは，入力サンプルが空間全体に一様に分布する場合， Delaunay Triangulation に収束することが証明されている [47]．Delaunay Triangulation は Voronoi 領域の隣接関係を表すネットワークである．Voronoi 領域とは，点集合に対して，どの点に最も近いかで空間を領域分割した各領域のことである．図 3.3a に例を示す．点集合 $\{\boldsymbol{w}_i\}_{i=1}^N$ に対する Voronoi 領域 $\{V_i\}_{i=1}^N$ は

$$V_i = \{\boldsymbol{x} \in \mathbb{R}^d \mid \forall k \neq i \ ||\boldsymbol{x} - \boldsymbol{w}_i|| \leq ||\boldsymbol{x} - \boldsymbol{w}_k||\}$$

である．この $\boldsymbol{w}_i$ が Delaunay Triangulation の各ノードに対応する (図 3.3b)．

最も近い 2 点が $\boldsymbol{w}_i, \boldsymbol{w}_j$ である点の集合

$$V_{ij} = \{\boldsymbol{x} \in \mathbb{R}^d \mid \forall k \neq i, j \ ||\boldsymbol{x} - \boldsymbol{w}_i|| \leq ||\boldsymbol{x} - \boldsymbol{w}_k|| \wedge ||\boldsymbol{x} - \boldsymbol{w}_j|| \leq ||\boldsymbol{x} - \boldsymbol{w}_k||\}$$

を second-order Voronoi 領域と言う．任意の Voronoi 領域 V_i, V_j が隣接している場合，$V_{ij} \neq \emptyset$ であり，隣接していない場合，$V_{ij} = \emptyset$ となる．アルゴリズム 2 より，Competitive Hebbian Learning では V_{ij} にサンプルが入力された場合に i, j 間にエッジが生成される．つまり，入力サンプルの分布を $p(\boldsymbol{x})$ とするなら

$$\int_{V_{ij}} p(\boldsymbol{x}) d\boldsymbol{x} > 0 \tag{3.5}$$

の場合に i, j 間にエッジが生成される．したがって，入力サンプルが空間全体に一様に分布し、十分なサンプルが与えられた場合，全ての隣接する Voronoi 領域対 V_i, V_j に対応する点 $\boldsymbol{w}_i, \boldsymbol{w}_j$ 間にエッジが形成され，Delaunay Triangulation となる．

入力サンプルがある多様体 $\mathcal{M}$ に沿って分布している場合，Competitive Hebbian Learning によって形成されるネットワークは，Delaunay Triangulation の誘導グラフとなり，$\mathcal{M}$ に沿った部分にのみエッジが形成される (図 3.3c)．この場合，V_i, V_j が隣接しており $V_{ij} \neq \emptyset$ であっても，V_{ij} が $\mathcal{M}$ と共通部分を持たない場合，

$$\int_{V_{ij}} p(\boldsymbol{x}) d\boldsymbol{x} = 0$$

となり，エッジが形成されない．このため，サンプルのない多様体以外の空間にはエッジは形成されず，多様体に沿ったネットワークが形成される．

SOINN ではこの Competitive Hebbian Learning に基づいてネットワークを

形成しており，そのネットワークはサンプルの多様体に沿った形になる．つまり，サンプルの分布の形状を表しており、あるノードの周りのエッジは，そのノードの周りの分布がどの方向にどのように広がっているかを表していると言える．

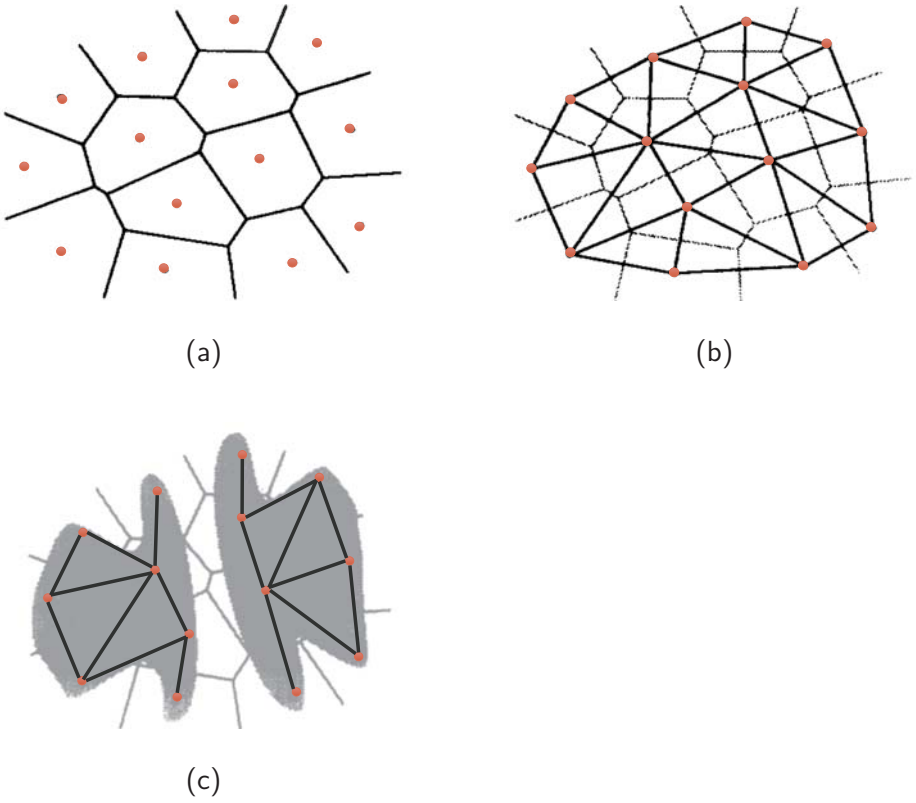


図 3.3: Competitive Hebbian Learning によって形成されるネットワークの例: (a) 与えられた点に対する Voronoi 領域 (b) サンプルが一様に分布する場合，Delaunay Triangulation に収束 (c) サンプルがある多様体に沿って分布する場合，(灰色の部分が多様体を表している．) Delaunay Triangulation の誘導グラフを形成し，多様体に沿った部分にのみエッジが形成される．

3.2.2　プロトタイプとしてのノード

第 3.2.1 章 で示した方法によって形成されたネットワークは，ノードがサンプルの多様体上にある場合に効果的に機能する．

SOINN では各ノードは競合学習において勝ち取ったサンプルを代表しており，サンプルの多様体上に位置している．これは，第 1 勝者ノードの更新，アルゴリズム 1 の 18 行目が，平均のオンライン更新と一致することから分かる．$\{x_1, x_2, \cdots, x_n\}$ の平均を

$$\boldsymbol{w}^{(n)} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i$$

とすると，$\{x_1, x_2, \cdots, x_{n+1}\}$ の平均 $\boldsymbol{w}^{(n+1)}$ は

$$\begin{aligned} \boldsymbol{w}^{(n+1)} &= \frac{1}{n+1}(n\boldsymbol{w}^{(n)} + \boldsymbol{x}_{n+1}) \\ &= \boldsymbol{w}^{(n)} + \frac{1}{n+1}(\boldsymbol{x}_{n+1} - \boldsymbol{w}^{(n)}) \end{aligned} \tag{3.6}$$

と，$\boldsymbol{w}^{(n)}$ と $\boldsymbol{x}_{n+1}$ から計算でき，この式はアルゴリズム 1 の 18 行目と一致する．つまり，各ノードは競合学習において勝ち取ったサンプルの平均に位置しており，それらを代表していると言える．アルゴリズム 1 の 19 行目はスムージングのためであり，$\psi_2(t)$ の係数は実験的に決められたものである．

また，アルゴリズム 1 の 18 行目 は k-Means 法 [53] を確率的勾配法 (Stochastic Gradient Descent: SGD) によってオンライン最適化する際の更新式と一致する [39]．したがって，アルゴリズム 1 の 18 行目だけを見れば，k-Means 法と

同じ目的関数

$$Q = \sum_{i=1}^{N} \min_{j} ||\boldsymbol{x}_i - \boldsymbol{w}_j||^2$$

を最小化していると考えられる．しかし，SOINN においてはプロトタイプの数が増減するため，完全に一致するわけではない．

SOINN のノードはオンラインに修正されていくため，修正前のエッジが適切でなくなる場合がある．そのため edge aging scheme [47] を用いて適切でなくなったエッジを削除する．各エッジは年齢というパラメータを保持している．Competitive Hebbian Learning によって新規に生成されたエッジの年齢は 1 に設定される．また，既存のエッジで Competitive Hebbina Learning によって更新されたエッジも年齢が 1 にリセットされる (アルゴリズム 1 の 15 行目)．第 1 勝者に接続しているが，更新されなかったエッジの年齢はインクリメントされていく (アルゴリズム 1 の 16 行目)．こうすることで，年齢が高いエッジは，接続しているノードはよく更新されているにも関わらず，あまり更新されていないエッジということになる．従って，年齢が閾値，エッジ削除ハイパーパラメータ age_{max} 以上になった場合，不適切なエッジと判断し削除する (アルゴリズム 1 の 20 行目)．

以上をまとめると，ノードは周辺のサンプルを代表するプロトタイプであり，そのノードの周辺ネットワークはノードの周辺サンプルの広がりの大きさとその方向を表していると考えられる．

4 自己増殖型ニューラルネットワークに基づくノンパラメトリック密度推定法

4.1 概要

第 3 章 で解説した SOINN の統計的な性質を利用することで，頑健性を有し高速なオンライン学習が可能なノンパラメトリック密度推定法 KDESOINN を提案する．

提案手法の概略図を図 4.1 に示す．サンプルのプロトタイプをノードとするネットワークとしてサンプル集合を学習し，そのネットワークの局所構造を利用して各ノードに配置するカーネルの形状と大きさを決定し，カーネルの線形和として密度関数を推定する．

4.2 推定密度関数

提案手法では第 3 章で述べた SOINN のネットワーク構造が表現するサンプルの分布の情報を用いて，密度推定を行う．

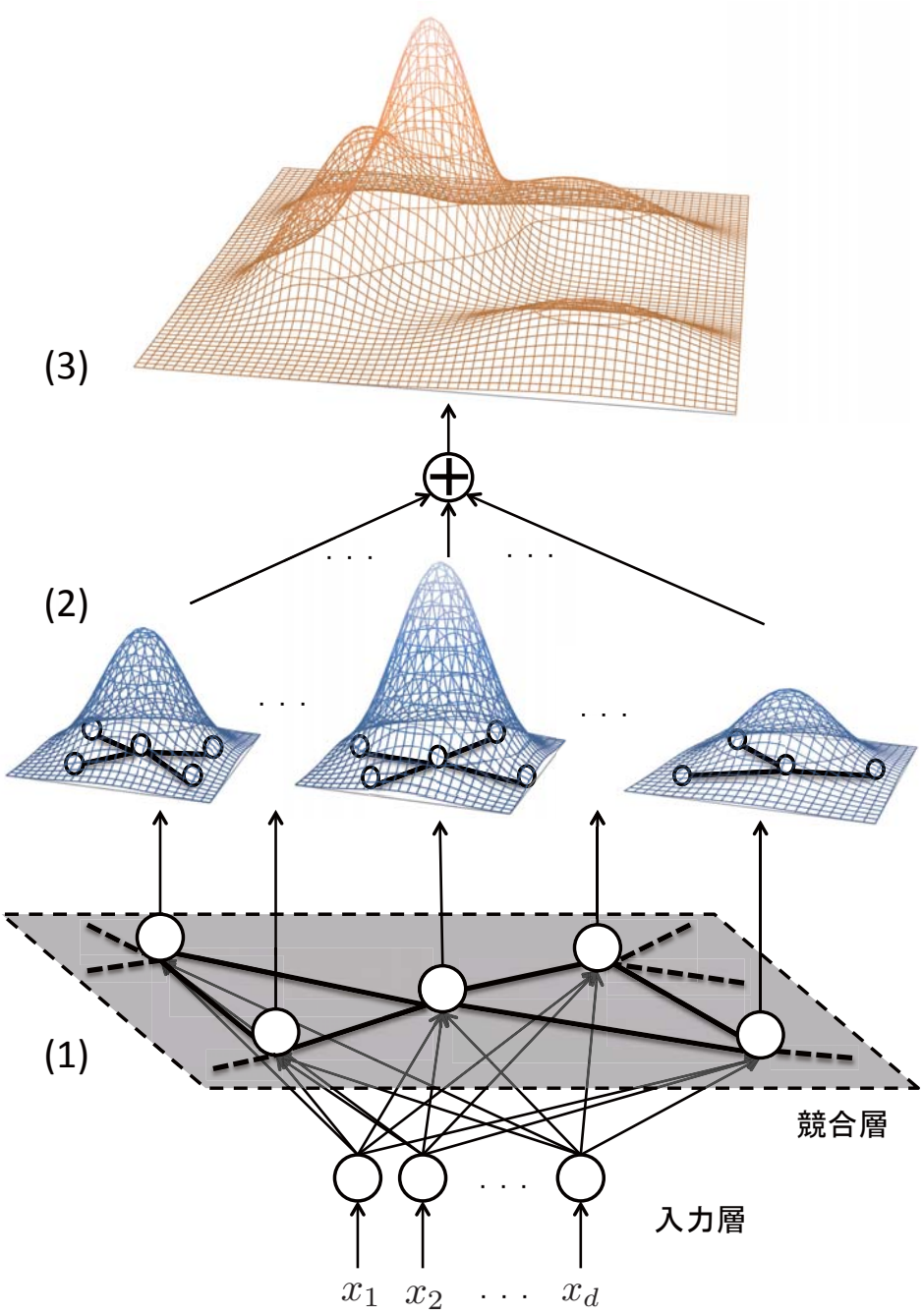


図 4.1: 提案手法の概略図：(1) サンプルのプロトタイプがノードであるネットワークとして，サンプルをオンラインに学習. (2) ネットワークの局所構造を利用して各ノードに配置するカーネルの形状と大きさを決定. (3) 各カーネルの線形和として密度関数を推定.

提案手法の推定密度関数 $\hat{p}(\boldsymbol{x})$ を以下のように定義する：

$$\hat{p}(\boldsymbol{x}) \quad = \quad \frac{1}{T_{\mathcal{N}}} \sum_{n \in \mathcal{N}} t_n K_{\boldsymbol{C}_n}(\boldsymbol{x}, \boldsymbol{w}_n), \tag{4.1}$$

ここで $T_{\mathcal{N}} = \sum_{n \in \mathcal{N}} t_n$，$K$ はガウスカーネル

$$K_{\boldsymbol{H}}(\boldsymbol{x}, \boldsymbol{\mu}) = \frac{1}{\sqrt{(2\pi)^d |\boldsymbol{H}|}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{H}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right)$$

である．各ノードの位置にガウスカーネルを配置して，その線形和としてサンプル集合の確率密度を推定する．ノード n に配置したカーネルは，ノード n が代表しているサンプルの数を表している t_n によって重み付けされている．

ガウスカーネルの形状を決定する共分散行列 $\boldsymbol{C}_n$ は各ノードの周辺のネットワーク構造から適応的に決定される．この共分散行列を局所ネットワーク共分散行列 (Local Network Covariance Matrix) と定義する：

$$\boldsymbol{C}_n \quad = \quad \frac{1}{T_{\mathcal{P}_n}} \sum_{p \in \mathcal{P}_n} t_p (\boldsymbol{w}_p - \boldsymbol{w}_n)(\boldsymbol{w}_p - \boldsymbol{w}_n)^T. \tag{4.2}$$

ここで，$T_{\mathcal{P}_n} = \sum_{i \in \mathcal{P}_n} t_i$ である．局所ネットワーク共分散行列は，共分散行列を拡張したものである．通常の共分散行列と異なる点は，中心が平均ではなく対象ノード n の位置ベクトルである点と，対象ノード n の隣接ノードのみで計算している点，各ノードの勝者として選択された回数 t で重み付けしている点である．n の隣接ノードが代表しているサンプルが，n から隣接ノードまでの距離と方向に広がっているとして，n の周辺のサンプルの広がりをこの局所ネットワーク共分散行列で近似しており，これをノードを中心とするガウスカーネルに組み込むことで，周辺のサンプルの密度を近似する．なお，ネットワークの形状によっては $\boldsymbol{C}_n$ が正則にならない場合があるが，その場合は対角成分に

微小な値を加える事で補正している．

4.3　学習アルゴリズム

アルゴリズム 3 に提案手法の学習アルゴリズムを示す．SOINN からの変更の 1 つに閾値領域に関する部分がある．この部分では，ノードとエッジの追加の基準を決めるため，SOINN のアルゴリズムにおいて最も重要な部分であると言える．

SOINN の閾値領域は超球であり，全ての方向に等距離に広がっており，これが分布を表すネットワークの形成に悪影響を及ぼすと考えられる．第 3 章 で述べたように，SOINN ではエッジが存在する方向にサンプルが存在しているが，閾値領域はエッジのない方向にも広がっている．このため，サンプルの密度を表さない長いエッジの形成される可能性がある．特に学習サンプルにノイズが含まれる場合，顕著であると考えられる．このエッジが分布を表すネットワークの形成に悪影響を及ぼすと考えられる．

これは，第 5 章 における改良と同じ動機からであり，詳しくは 第 5 章 に譲る．

ここでの変更はヒューリスティックスであり十分ではない．第 5 章 では，よりネットワークの性質を考慮した改良を加えることで性能向上を実現している．

閾値領域決定以外にも適切なネットワークを形成するように変更を加えている．

SOINN のアルゴリズム (アルゴリズム 1) における 19 行目は廃止している．これはクラスタリングのためのスムージングが目的で実験的に決められたものであり，確率密度推定には必要ないと考えられるためである．クラスタリングにおいては，他クラスタから分離するという目的で，同一クラスタ内のノード間の距離を近づけることが有効に働いた．しかし，提案手法においては，第 3.2.2

章で示したように，ノードは競合学習で勝ち取ったサンプルの平均に位置し，式 (4.1) より，各ノード周辺のサンプルは局所ネットワーク共分散行列によって規定されるガウス分布によって近似されるので，このようなスムージングは必要ないと考えられる．

28 行目では $|\mathcal{P}_i| \leq 1$ ではなく $|\mathcal{P}_i| = 0$ としている．SOINN においては，ネットワークがクラスタを表しており，ノイズの影響を抑えるためや、クラスタを分離するために，なるべく不要なノードは削除するようになっている．しかし，密度推定において，ネットワークは分布の広がりを表しており，ノードを削除する条件を強くする必要はないと考えられるためである．

10 から 15 行目において，第一勝者ノード s_1 に近い入力サンプルは，s_1 の更新にのみ使われるようにしている．通常の SOINN (アルゴリズム 1) では，入力サンプルが s_1 に近くても s_2 の閾値領域内に入らなければ，そのサンプルは新規ノードとして追加される．これがネットワークの形成に悪影響を与えているのではないかと考えられるため，10 から 15 行目において，s_2 の閾値領域内に入っていなくても，s_1 の更新に入力サンプルを使うようにしている．

アルゴリズム 3: KDESOINN の学習アルゴリズム

if 初回の学習である **then**

　$\mathcal{N} \leftarrow \{c_1, c_2\}$

　　▷ 学習データからランダムに選択したサンプルを位置ベクトルとして持つ 2 つのノード c_1, c_2 で全ノードの集合を初期化

　$\mathcal{E} \leftarrow \phi$

end if

while 入力パターン $\boldsymbol{\xi} \in \mathbb{R}^d$ が存在する **do**

　$s_1 \leftarrow \arg\min_{c \in \mathcal{N}} ||\boldsymbol{\xi} - \boldsymbol{w}_c||$

　　▷ $\boldsymbol{\xi}$ に対する第 1 勝者ノード s_1 を探索

　$s_2 \leftarrow \arg\min_{c \in \mathcal{N} \setminus \{s_1\}} ||\boldsymbol{\xi} - \boldsymbol{w}_c||$

　　▷ $\boldsymbol{\xi}$ に対する第 2 勝者ノード s_2 を探索

　次式によって類似度閾値 $\Theta_{s_1}, \Theta_{s_2}$ を計算:

$$
\begin{aligned}
\Theta_i &= \sqrt{\boldsymbol{d}^T \boldsymbol{M} \boldsymbol{d}}, \\
&\boldsymbol{d} = \frac{\boldsymbol{\xi} - \boldsymbol{w}_i}{||\boldsymbol{\xi} - \boldsymbol{w}_i||}, \\
&\boldsymbol{M}_i = \boldsymbol{C}_i + \rho \gamma_i \boldsymbol{I}, \\
&\gamma_i = \begin{cases} \min\limits_{p \in \mathcal{P}_i} ||\boldsymbol{w}_p - \boldsymbol{w}_i|| & (\mathcal{P}_i \neq \emptyset) \\ \min\limits_{p \in \mathcal{N} \setminus \{i\}} ||\boldsymbol{w}_p - \boldsymbol{w}_i|| & (otherwise). \end{cases}
\end{aligned}
$$

　if $||\boldsymbol{\xi} - \boldsymbol{w}_{s_1}|| > \Theta_{s_1}$ OR $||\boldsymbol{\xi} - \boldsymbol{w}_{s_2}|| > \Theta_{s_2}$ **then**

　　if $||\boldsymbol{\xi} - \boldsymbol{w}_{s_1}|| < \frac{\Theta_{s_1}}{4}$ **then**

　　　$t_{s_1} \leftarrow t_{s_1} + 1$ ▷ s_1 の勝者として選択された回数をインクリメント

12: $\boldsymbol{w}_{s_1} \leftarrow \boldsymbol{w}_{s_1} + \psi_1(t_{s_1})(\boldsymbol{\xi} - \boldsymbol{w}_{s_1})$ ▷ s_1 の位置ベクトルを更新

13: **else**

14: $\mathcal{N} \leftarrow \mathcal{N} \cup \{\boldsymbol{\xi}\}$ ▷ $\boldsymbol{\xi}$ を新規ノードとして追加

15: **end if**

16: **else**

17: **if** $(s_1, s_2) \notin \mathcal{E}$ **then**

18: $\mathcal{E} \leftarrow \mathcal{E} \cup \{(s_1, s_2)\}$ ▷ エッジ (s_1, s_2) を追加

19: **end if**

20: $a_{(s_1,s_2)} \leftarrow 0$ ▷ エッジ (s_1, s_2) の年齢を0にリセット

21: $a_{(s_1,i)} \leftarrow a_{(s_1,i)} + 1 \quad (\forall i \in \mathcal{P}_{s_1})$

▷ s_1 につながる全エッジの年齢をインクリメント

22: $t_{s_1} \leftarrow t_{s_1} + 1$ ▷ s_1 の勝者として選択された回数をインクリメント

23: $\boldsymbol{w}_{s_1} \leftarrow \boldsymbol{w}_{s_1} + \psi_1(t_{s_1})(\boldsymbol{\xi} - \boldsymbol{w}_{s_1})$ ▷ s_1 の位置ベクトルを更新

24: エッジ $\mathcal{E}_{old} = \{e \mid e \in \mathcal{E},\ a_e > age_{max}\}$ を削除

25: ノード $\{i \mid \exists j\ (i,j) \in \mathcal{E}_{old}, |\mathcal{P}_i| = 0\}$ を削除

▷ 24行目 において削除されたエッジに接続していたノードのうち，接続エッジがなくなったノードを削除.

26: **end if**

27: **if** 入力パターン数が λ の倍数 **then**

28: ノード $\{i \mid |\mathcal{P}_i| = 0\}$ を削除

29: **end if**

30: **end while**

4.4　評価実験

ロバスト性，学習時間，精度に関して提案手法の優位性を示すために評価実験を行った．全ての実験は Intel ® Core ™ i7-3770，16.0 GB RAM を搭載した計算機上で MATLAB を用いて行った．

4.4.1　比較手法

既存の比較手法として，典型的な KDE に対して尤度を評価尺度とし交差検証法によりバンド幅行列を最適化するバッチ手法（CVML）[25]，KDE をカーネル法と捉え，ロバストな推定を行うことでロバストな KDE を実現したバッチ手法 (RKDE) [18]，オンラインクラスタリングを用いてカーネルの形状と数を適応的に決定するオンライン手法（oKDE）[28]，を用いた．RKDE は他の手法でバンド幅行列を求める必要があるため，CVML が最適化したバンド幅行列を RKDE のバンド幅行列として用いた．

4.4.2　ロバスト性

ロバスト性を評価するために，学習データにノイズが含まれる場合の推定精度を既存手法と比較した．学習データは

$$\boldsymbol{X} \sim p = (1-\alpha)\ p_{target} + \alpha\ p_{noise},$$

とする．p_{target} が真の分布，p_{noise} がノイズの分布であり，α はノイズの割合を表す．各手法は $\boldsymbol{X}$ を学習し，p_{target} を推定する．α を変化させ，それに伴う

真の分布の推定精度の変化を評価した．真の分布には

$$p_{target}(\boldsymbol{x}) = N(\boldsymbol{x}; \boldsymbol{t}, 0.2 \cdot \boldsymbol{I}), \quad (4.3)$$
$$\boldsymbol{t} = [a, \sin(3a)]^T, a \in [-2, 2],$$

を用い，ノイズの分布 p_{noise} には一様分布（各次元の範囲を $[-4, 4]$ とした）を用いた．学習サンプル数は 5,000 とした．図 4.2 に学習サンプルの例を図示する．

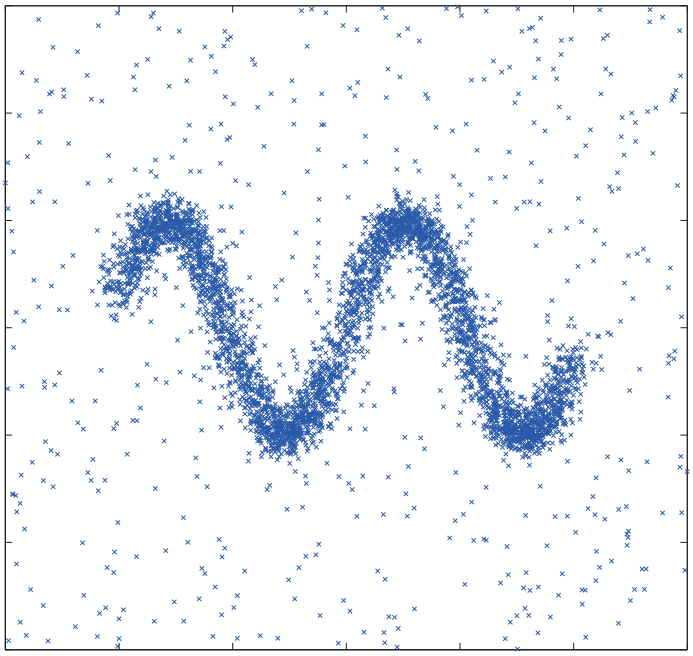


図 4.2: ロバスト性に関する評価実験の学習データの例：式 (4.3) の分布からのサンプル集合にノイズを 20% 含めた学習データの例.

評価尺度には Jensen—Shannon (JS) divergence を用いた．JS divergence は確率分布間の距離尺度であり，推定分布と真の分布を比較した場合，値が低いほど推定精度が高いといえる．確率分布間の距離尺度としては Kullback-Leibler (KL) divergence がより一般的だが，KL divergence は対称でないため，真の分

布または推定分布のいずれかの領域のみしか評価できないため，ロバスト性の評価尺度には適していない．ノイズの影響により，真の分布と推定分布の領域は一致しない可能性があり，KL divergence を用いると一致しなかった部分が評価されないためである．JS divergence は KL divergence と違い，対称でありロバスト性の評価に適している．また，無限大に発散することもない．推定確率分布 $\hat{p}$ と p_{target} 間の JS divergence $D_{JS}(\hat{p}||p_{target})$ は

$$D_{JS}(\hat{p}||p_{target}) = \frac{1}{2}D_{KL}(\hat{p}||m) + \frac{1}{2}D_{KL}(p_{target}||m)$$

である．ここで $m = \frac{1}{2}(\hat{p} + p_{target})$ ，D_{KL} は KL divergence であり，

$$D_{KL}(p||q) \approx \frac{1}{n}\sum_{i=1}^{n}\log\frac{p(x_i)}{q(x_i)}$$

である．ここで $\{x_i\}_{i=1}^{n}$ は p からのサンプルである．よって，

$$\begin{aligned} &D_{JS}(\hat{p}||p_{target}) \\ &\approx \frac{1}{2n}\sum_{i=1}^{n}\log\frac{\hat{p}(x_{\hat{p}i})}{m(x_{\hat{p}i})} + \frac{1}{2n}\sum_{i=1}^{n}\log\frac{p_{target}(x_{p_{target}i})}{m(x_{p_{target}i})} \end{aligned}$$

となる．ここで，$\{x_{\hat{p}i}\}_{i=1}^{n}$ は $\hat{p}$ からのサンプルであり，$\{x_{p_{target}i}\}_{i=1}^{n}$ は p_{target} からのサンプルである．今回の実験においては，$n = 20,000$ とした．各ノイズ割合に対して 20 回ずつ実験を行い，その平均を評価した．

oKDE のハイパーパラメータを $D_{th} = 0.01$ とした．提案手法のハイパーパラメータは $\lambda = 300, age_{max} = 50, \rho = 0.1$ とした．

図 4.3 が実験結果である．学習データにおけるノイズ割合が 0 %の場合では，提案手法は既存手法とほぼ同等の精度を出している．ロバスト性のない既存手法である CVML と oKDE は学習データにおけるノイズの割合が増加するに

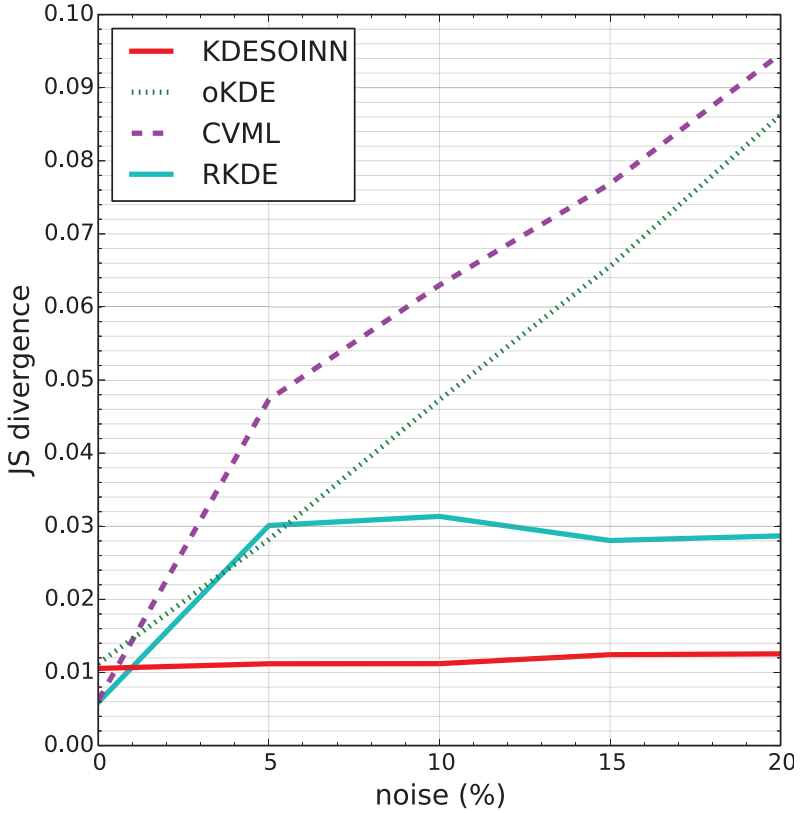


図 4.3: ロバスト性に関する評価実験の結果：横軸が学習データにおけるノイズの割合 α，縦軸が推定精度の評価尺度である JS divergence を表している．

従って大きく精度が低下している．特に CVML はノイズの割合が 0 %から 5 %になった時の精度の低下が大きいことからノイズに敏感に反応してしまうことが分かる．oKDE はカーネルの形状，大きさ，および数をデータに合わせて適応的に設定するため CVML 程は敏感には反応していないと考えられる．しかし，それだけではロバスト性は十分ではなくノイズの割合が増加するに従って精度が低下している．これに対して，ロバスト性のある提案手法と RKDE はノイズが増加しても精度が低下しない．特に提案手法は 0 %の時と同程度の水準を保っている．このことから提案手法が高いロバスト性を有していると示さ

れた.

4.4.3　学習時間

学習データ数に対する学習時間の変化を評価した．ノイズを含まない 式 (4.3) の分布からのサンプルを用いて，データ数を 1,000 から 100,000 まで増加させたときの計算時間の変化を既存手法と比較した．各手法のパラメータは 第 4.4.2 章と同じとした.

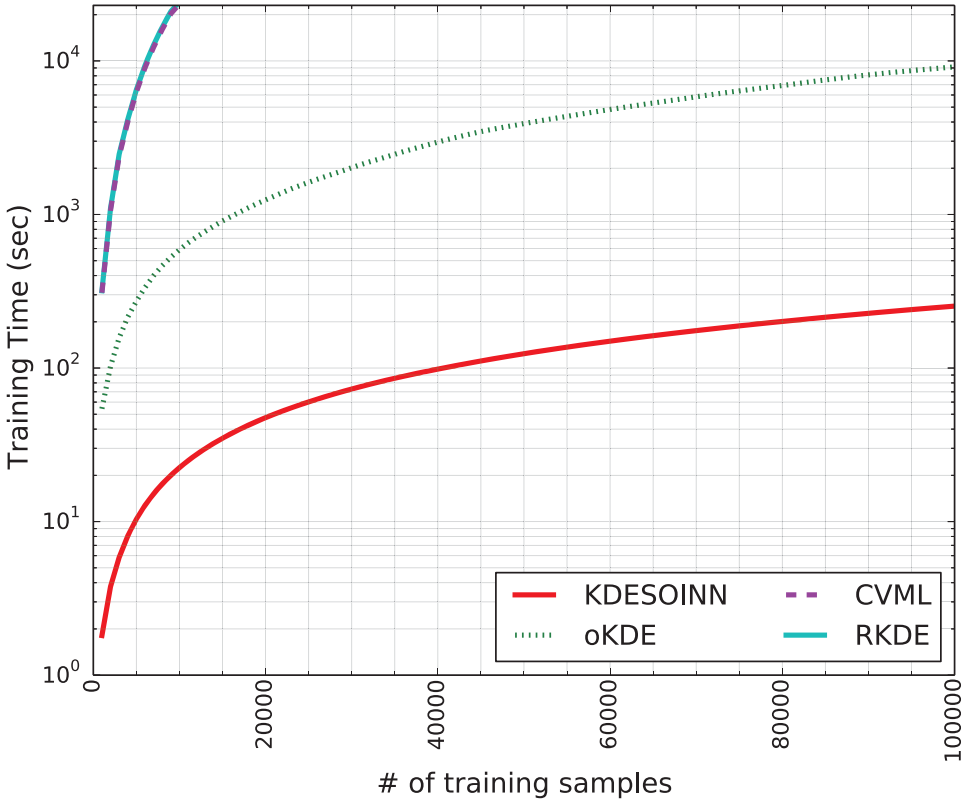


図 4.4: 学習時間に関する評価実験の結果：学習サンプル数が 1,000 から 100,000 まで変化させた時の学習時間の変化を示している．縦軸が対数表示であることに注意されたい.

実験結果を図 4.4 に示す．CVML 及び RKDE は非常に学習に時間が掛かるため，学習サンプル数が 10,000 の時点で実験を中止した．CVML はバッチ手法であり，学習サンプル数の増加に伴って学習時間が劇的に増加している．oKDE と提案手法はオンライン手法であるため CVML ほど学習時間はかかっていないが，oKDE は学習サンプル数が増加すると共に学習時間が大きく増加しているのに対し，提案手法は増加が小さい．oKDE はクラスタリングを用いて混合数を減らし，バンド幅行列を入力サンプルごとに計算し直しているのに対し，提案手法はプロトタイプのネットワーク構造の中にカーネルの配置位置とバンド幅行列の情報を含めているため，より高速に実行できると考えられる．

4.4.4　実データによる精度評価

表 4.1: 第 4.4.4 章 における精度評価で用いた実データセット一覧

	サンプル数	次元数	クラス数
Iris	150	4	3
Wine	178	13	3
Pima	768	8	2
Wine_Red	1,599	11	6
Wine_White	4,898	11	7

UCI Machine Learning Repository のデータセットを使って実データにおける確率密度推定を評価した．実験に用いたデータセットを表 4.1 に示す．データセットのクラスごとに実験を行った．各実験のサンプル集合を白色化により標準化し，そこからランダムに抽出した 75 %のサンプルを学習データ，残りをテストデータとし，密度関数の推定精度を評価した．これを各実験で 20 回繰り返し，データセットごとに平均し評価した．評価尺度には Negative Log Likelihood (NLL) を用

いた．NLL は無限大に発散する可能性があるため，順序平均をとることで対応した．提案手法のハイパーパラメータを $\lambda = 300, age_{max} = 50, \rho = 0.5, D_{th} = 0.1$，oKDE のハイパーパラメータを $D_{th} = 0.1$ とした．

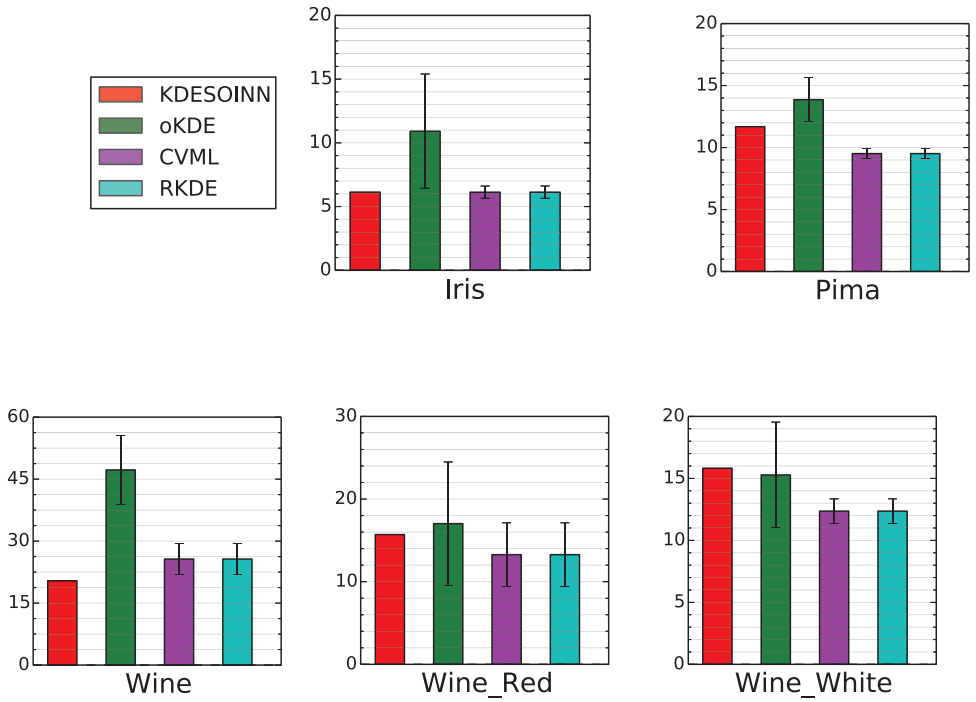


図 4.5: 実データに対する密度推定の実験結果：各グラフは各データセットにおける各手法の推定精度を表す．評価尺度は negative log likelihood (NLL), 棒グラフが平均，エラーバーが標準偏差を表す．

実験結果を 図 4.5 に示す．棒グラフは平均値，それに付随するバーは標準偏差を表している．幾つかのデータセットにおいて提案手法は既存のバッチ手法と同程度の精度を出しているが，他のデータセットでは劣っている．オンライン手法の oKDE と比較すると，提案手法は殆どのデータセットで優れた精度を出している．また，提案手法の標準偏差は他手法よりも小さく，安定性が高い

と言える．

4.5　考察

4.5.1　提案推定密度関数の有効性の検証

ここでは，提案手法の推定密度関数が SOINN のネットワークの統計的な性質を如何に引き出しているかを，単純に SOINN のネットワークの各ノードに対してガウスカーネルを配置する場合と比較することで検証したい．

第 4.4.2 章と同様の設定で実験を行った．比較する手法は次の 2 手法である．学習を SOINN で行い，学習したノードの重みベクトルを用いて CVML と同様に尤度交差検証法を行い，各ノードに配置するカーネルのバンド幅行列を決定している手法 (Naive) と，SOINN で学習したネットワーク構造に対して 式 (4.1) を用いて推定密度関数を算出する手法 (Adaptive) である．各種法において学習アルゴリズムのハイパーパラメータは $\lambda = 500, age_{max} = 100, \rho = 0.1, k = 2d$ とした．

図 4.6 が実験の結果である．Adaptive の方が全体を通して精度が高く，特にノイズの割合が増加すると顕著になる．SOINN はノードの密度だけではなく，エッジやノード，勝者として選択され回数などネットワークとしてサンプルの分布を保持しており，提案手法の推定密度関数 式 (4.1) がこのネットワークが保持している密度の情報を引き出せていることが分かる．また，Naive は通常の KDE と同様に各ノードに配置しているカーネルは全て同じであるのに対し，Adaptive では 式 (4.1)，式 (4.2) を用い，各ノードに配置するカーネルの形状および大きさをネットワークから適応的に決定しているためロバスト性が向上し，ノイズの割合が増加すると性能の差が顕著になったと考えられる．

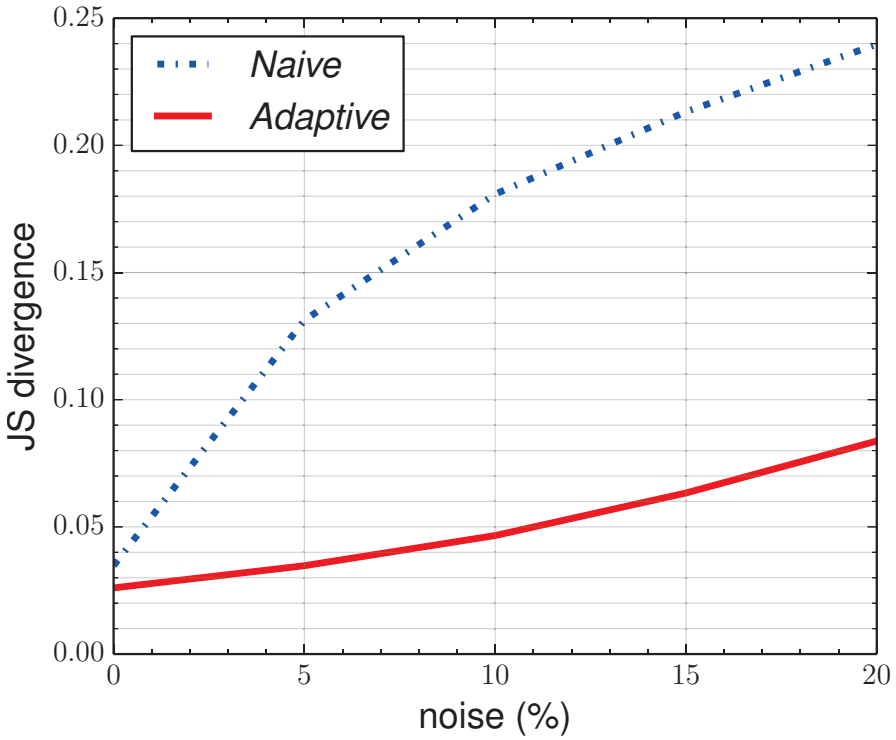


図 4.6: ロバスト性に関する提案推定密度関数の検証実験の結果

5 学習アルゴリズムの改良による性能向上

第 4 章 では SOINN のネットワークを基にノンパラメトリックな密度推定法を考案した．ここでは，ネットワーク形成の仕組みを考察することで，更なる性能向上を試みる．

5.1 SOINN における Competitive Hebbian Learning と閾値領域

Competitive Hebbian Learning においては，第 3.2.1 章 で述べたように，ある 2 ノードの second-order Voronoi 領域に入力サンプルが 1 つでも入れば，それらのノード間にエッジが形成される．second-order Voronoi 領域はノードが疎な領域では大きくなる傾向がある．したがって，学習データにノイズが含まれている場合，所望の分布とは関係のないエッジが形成される可能性が高くなる．つまり， Competitive Hebbian Learning によって形成されるネットワークはロバスト性がないと言える．

SOINN では閾値領域という制約を導入することで，Competitive Hebbian Learning にロバスト性を実現している．Competitive Hebbian Learning では，

入力サンプルに対する最近傍 2 ノード (第 1, 第 2, 勝者ノード) 間にエッジを張るのに対し，SOINN では，第 1, 第 2 勝者ノードそれぞれの閾値領域に入力サンプルが含まれる場合にのみエッジを張る．また，この閾値領域はノード及びエッジの追加の基準にもなっている．アルゴリズム 1 の 10 行目 において，エッジの追加に使われなかった入力サンプルは，勝者ノードが代表しているサンプルではないとして，ノードとして追加される．

SOINN の閾値領域は各ノードを中心とする超球である．アルゴリズム 1 より，ノード i の閾値領域は，あるサンプル $\boldsymbol{\xi}$ に対して

$$||\boldsymbol{\xi} - \boldsymbol{w}_{s_1}|| \leq \Theta_i \tag{5.1}$$

を満たす領域とする．ここで

$$\Theta_i = \begin{cases} \max\limits_{p \in \mathcal{P}_i} ||\boldsymbol{w}_p - \boldsymbol{w}_i|| & (\mathcal{P}_i \neq \phi) \\ \min\limits_{p \in \mathcal{N} \setminus \{i\}} ||\boldsymbol{w}_p - \boldsymbol{w}_i|| & (otherwise). \end{cases} \tag{5.2}$$

である．つまり，SOINN における各ノードの閾値領域は，隣接ノードがある場合は，最遠方隣接ノードまでの距離を半径とする超球，隣接ノードがない場合は，最近傍ノードまでの距離を半径とする超球である．

SOINN では，各ノード周辺のサンプルは等方的ガウス分布に従うと仮定している．勝者ノードの閾値領域内に入った入力サンプルのみが勝者ノードが代表しているサンプルとしてカウントされる．したがって，閾値領域の形状がノード周辺のサンプルの分布の形状を表しているといえる．SOINN においては閾値領域の形状は超球であり，各ノードの周辺のサンプルの分布は等方的ガウス分布，共分散行列が単位行列の定数倍で分布の広がりが方向に依存しないガウス

分布，であると仮定している．

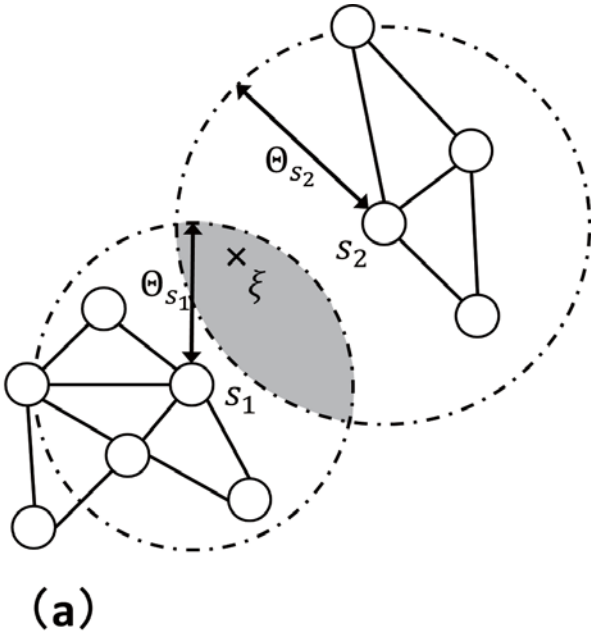
これは Competitive Hebbian Learning によって形成されたネットワークと矛盾している．Competitive Hebbian Learning によって形成されるネットワークは，入力サンプルがある多様体に沿って分布している場合，多様体に沿った部分にのみエッジが形成される．つまり，入力サンプルはネットワークに沿って分布しているということである．したがって，あるノードの周辺のサンプルは，そのノードから出ているエッジの方向に広がっており，全ての方向に同じように広がっているわけではない．

そのため，SOIINN では密度を表さないエッジが形成されてしまう．図 5.1a に示すように，SOINN の閾値領域はエッジのない方向，分布が広がっていない方向，にも広がっており，ここにノイズが入力されると分布を表さないエッジが形成されてしまうのである．

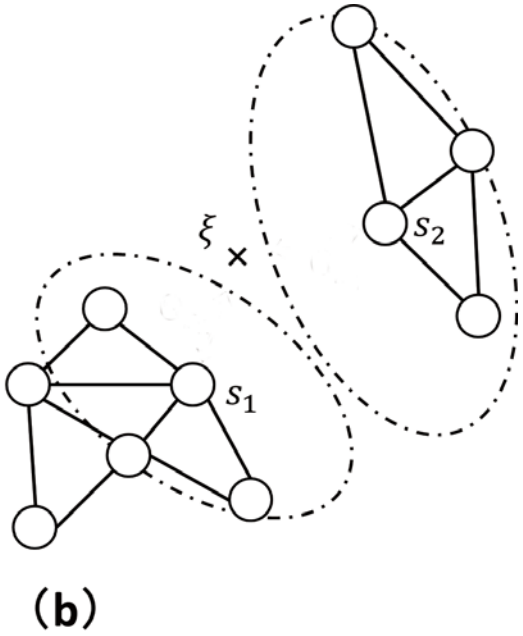
5.2　閾値領域の改良

第 5.1 章 で述べてた問題点を解決するために，閾値領域に改良を加えた．

Competitive Hebbian Learning によって形成されたネットワークに沿ってサンプルは分布しており，各ノード周辺のサンプルは，そのノードに接続するエッジに沿って分布している．つまり，局所ネットワーク共分散行列が規定するガウス分布に従っていると言える．なぜなら，局所ネットワーク共分散行列が規定するガウス分布は，ノード周りの局所的なネットワークからノード周辺のサンプルの分布を近似するように設計されているためである．



(a) SOINN の閾値領域



(b) 提案手法の閾値領域

図 5.1: SOINN と提案手法の閾値領域の比較

そこで，閾値領域を局所ネットワーク共分散行列が規定するガウス分布の形状になるように改良した．ノード i の閾値領域は，あるサンプル $\boldsymbol{\xi}$ に対して

$$(\boldsymbol{\xi} - \boldsymbol{w}_i)^T \boldsymbol{M}_i^{-1} (\boldsymbol{\xi} - \boldsymbol{w}_i) \leq 1,$$

を満たす領域とする．ここで

$$\begin{aligned} \boldsymbol{M}_i &= \boldsymbol{C}_i + \rho\gamma_i \boldsymbol{I}, \qquad (5.3) \\ \gamma_i &= \begin{cases} \min\limits_{p \in \mathcal{P}_i} ||\boldsymbol{w}_p - \boldsymbol{w}_i|| & (\mathcal{P}_i \neq \emptyset) \\ \min\limits_{p \in \mathcal{N} \setminus \{i\}} ||\boldsymbol{w}_p - \boldsymbol{w}_i|| & (otherwise) \end{cases} \end{aligned}$$

である．単位行列を加えているのは，スムージングのためである．

図 5.1 において，SOINN の閾値領域と提案手法の閾値領域を比較している．SOINN の閾値領域は超球であり，ネットワークが無い方向にも広がっているのに対し，提案手法の閾値領域は局所ネットワーク共分散行列が規定するガウス分布の形状をしており，ネットワークに沿うように広がっている．SOINN ではサンプルの分布とは関係のないエッジになっていた入力サンプル $\boldsymbol{\xi}$ は，閾値領域に入らずにノードとなり，その後の入力サンプルによってネットワークに組み込まれるか，ノイズとして削除されるか決定される．このようにすることで，分布を表さないネットワークの形成を抑えることができる．

5.3　その他の改良

閾値領域決定以外にも適切なネットワークを形成するように変更を加えている．23-24，27-28 行目 ではネットワークの調整を行っている．SOINN ではノー

ド間の閾値領域内にサンプルが入力されないとエッジが作成されないため，高次元空間やサンプル数が少ない場合，十分な数のエッジが作成されない．これを補うために，ここでは k 近傍グラフに基づいてエッジを追加している．ノード集合に対して k 近傍グラフを形成し，そのグラフにおいて双方向にエッジが存在するノードのペアに対して，エッジを追加する．ノードが密に分布する場所では k 近傍グラフのエッジは双方向に形成されるので，密度を表すエッジが張れると考えられる．

アルゴリズム 4 に改良後の学習アルゴリズムの全体を示す．

アルゴリズム 4: 改良 KDESOINN の学習アルゴリズム

---

if 初回の学習である **then**

　$\mathcal{N} \leftarrow \{c_1, c_2\}$

　　▷ 学習データからランダムに選択したサンプルを位置ベクトルとして持つ 2 つのノード c_1, c_2 で全ノードの集合を初期化

　$\mathcal{E} \leftarrow \phi$

end if

while 入力パターン $\boldsymbol{\xi} \in \mathbb{R}^d$ が存在する **do**

　$s_1 \leftarrow \arg\min_{c \in \mathcal{N}} ||\boldsymbol{\xi} - \boldsymbol{w}_c||$

　　▷ $\boldsymbol{\xi}$ に対する第 1 勝者ノード s_1 を探索

　$s_2 \leftarrow \arg\min_{c \in \mathcal{N} \setminus \{s_1\}} ||\boldsymbol{\xi} - \boldsymbol{w}_c||$

　　▷ $\boldsymbol{\xi}$ に対する第 2 勝者ノード s_2 を探索

　if $(\boldsymbol{\xi} - \boldsymbol{w}_{s_1})^T \boldsymbol{M}_{s_1}^{-1} (\xi - \boldsymbol{w}_{s_1}) > 1$ OR $(\boldsymbol{\xi} - \boldsymbol{w}_{s_2})^T \boldsymbol{M}_{s_2}^{-1} (\boldsymbol{\xi} - \boldsymbol{w}_{s_2}) > 1$ **then**

9: $\mathcal{N} \leftarrow \mathcal{N} \cup \{\boldsymbol{\xi}\}$ ▷ $\boldsymbol{\xi}$ を新規ノードとして追加

10: **else**

11: **if** $(s_1, s_2) \notin \mathcal{E}$ **then**

12: $\mathcal{E} \leftarrow \mathcal{E} \cup \{(s_1, s_2)\}$ ▷ エッジ (s_1, s_2) を追加

13: **end if**

14: $a_{(s_1,s_2)} \leftarrow 0$ ▷ エッジ (s_1, s_2) の年齢を 0 にリセット

15: $a_{(s_1,i)} \leftarrow a_{(s_1,i)} + 1 \ \ (\forall i \in \mathcal{P}_{s_1})$
▷ s_1 につながる全エッジの年齢をインクリメント

16: $t_{s_1} \leftarrow t_{s_1} + 1$ ▷ s_1 の勝者として選択された回数をインクリメント

17: $\boldsymbol{w}_{s_1} \leftarrow \boldsymbol{w}_{s_1} + \psi_1(t_{s_1})(\boldsymbol{\xi} - \boldsymbol{w}_{s_1})$ ▷ s_1 の位置ベクトルを更新

18: エッジ $\mathcal{E}_{old} = \{e \mid e \in \mathcal{E},\ a_e > age_{max}\}$ を削除

19: ノード $\{i \mid \exists j\ (i,j) \in \mathcal{E}_{old}, |\mathcal{P}_i| = 0\}$ を削除
▷ 18 行目 において削除されたエッジに接続していたノードのうち，接続エッジがなくなったノードを削除.

20: **end if**

21: **if** 入力パターン数が λ の倍数 **then**

22: ノード $\{i \mid |\mathcal{P}_i| = 0\}$ を削除

23: ノード集合が $\mathcal{N}$ である k-NN グラフ $\mathcal{G}$ を作成

24: $\mathcal{E} \leftarrow \mathcal{E} \cup \{(i,j) \mid (i,j) \in E(\mathcal{G}), (j,i) \in E(\mathcal{G})\}$

25: **end if**

26: **end while**

27: ノード集合が $\mathcal{N}$ である k-NN グラフ $\mathcal{G}$ を作成

28: $\mathcal{E} \leftarrow \mathcal{E} \cup \{(i,j) \mid (i,j) \in E(\mathcal{G}), (j,i) \in E(\mathcal{G})\}$

5.4　評価実験

改良 KDESOINN (improved KDESOINN: iKDESOINN) の改良の効果を確かめるために、評価実験を行った.

5.4.1　ロバスト性

ロバスト性を評価するために，第 4.4.2 章 と同様に学習データにノイズが含まれる場合の推定精度を既存手法と比較した．比較手法は 第 4.4.1 章 の既存手法と改良していない KDESOINN である．学習データは

$$\boldsymbol{X} \sim p = (1-\alpha)\ p_{target} + \alpha\ p_{noise},$$

とし，p_{target} が真の分布，p_{noise} がノイズの分布であり，α はノイズの割合を表す．各手法は $\boldsymbol{X}$ を学習し，p_{target} を推定する．α を変化させ，それに伴う真の分布の推定精度の変化を評価した．真の分布 p_{target} として次の Sine 形と螺旋形の 2 つの分布を用い，ノイズの分布 p_{noise} として一様分布とガウス分布を用いた．つまり，次の 4 つの場合で実験を行った.

$$\begin{aligned} p_{target}(\boldsymbol{x}) &= N(\boldsymbol{x}; \boldsymbol{t}, 0.2 \cdot \boldsymbol{I}), \\ & \quad \boldsymbol{t} = [a, \sin(3a)]^T, a \in [-2, 2], \\ p_{noise}(\boldsymbol{x}) &= U(\boldsymbol{x}; -4, 4). \end{aligned} \tag{5.4}$$

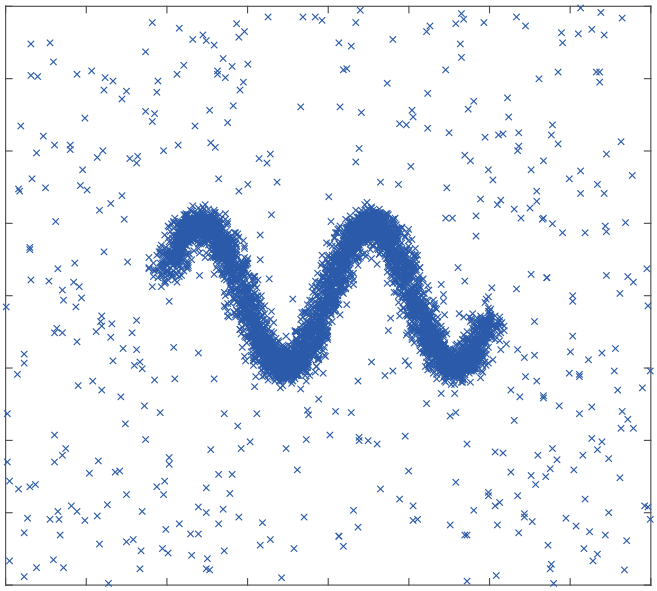
$$
\begin{aligned}
p_{target}(\boldsymbol{x}) &= N(\boldsymbol{x}; \boldsymbol{t}, 0.2 \cdot \boldsymbol{I}), \\
&\quad \boldsymbol{t} = [a, \sin(3a)]^T, a \in [-2, 2], \\
p_{noise}(\boldsymbol{x}) &= N(\boldsymbol{x}; [0, 3]^T, 2.0 \cdot \boldsymbol{I}).
\end{aligned} \tag{5.5}
$$

$$
\begin{aligned}
p_{target}(\boldsymbol{x}) &= N(\boldsymbol{x}; \boldsymbol{t}, 0.25 \cdot \boldsymbol{I}), \\
&\quad \boldsymbol{t} = [r\cos(\theta), r\cos(\theta), \theta]^T, \\
&\quad r = 10 - 0.5\theta, \theta \in [-7, 7], \\
p_{noise}(\boldsymbol{x}) &= U(\boldsymbol{x}; -20, 20).
\end{aligned} \tag{5.6}
$$

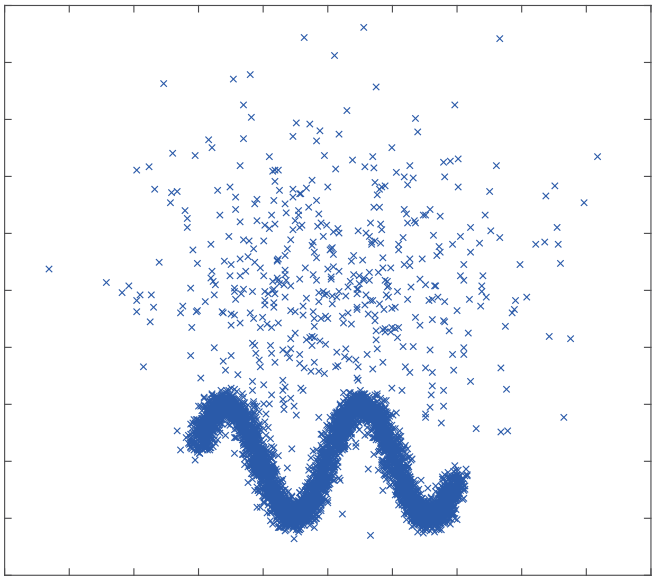
$$
\begin{aligned}
p_{target}(\boldsymbol{x}) &= N(\boldsymbol{x}; \boldsymbol{t}, 0.25 \cdot \boldsymbol{I}), \\
&\quad \boldsymbol{t} = [r\cos(\theta), r\cos(\theta), \theta]^T, \\
&\quad r = 10 - 0.5\theta, \theta \in [-7, 7], \\
p_{noise}(\boldsymbol{x}) &= N(\boldsymbol{x}; [0, 0, -15]^T, 30 \cdot \boldsymbol{I}).
\end{aligned} \tag{5.7}
$$

学習データの例を 図 5.2 及び 図 5.3 にそれぞれ図示する．学習サンプル数は 5,000 とした．各ノイズ割合に対して 20 回ずつ実験を行い，その平均を評価した．評価尺度には Jensen—Shannon (JS) divergence を用いた．

oKDE のハイパーパラメータを $D_{th} = 0.01$ とした．提案手法のハイパーパラメータは Sine 形の分布 (式 (5.4), 式 (5.5)) の場合，$\lambda = 500, age_{max} = 300, \rho = 0.1, k = 2d$，螺線形の分布 (式 (5.6), 式 (5.7)) の場合，$\lambda = 1000, age_{max} = 300, \rho = 1.0, k = 2d$ とした．

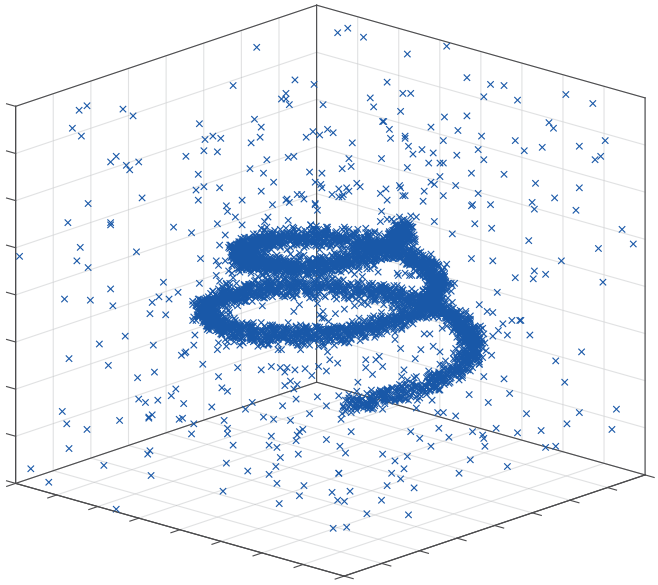


(a) 式 (5.4) の分布

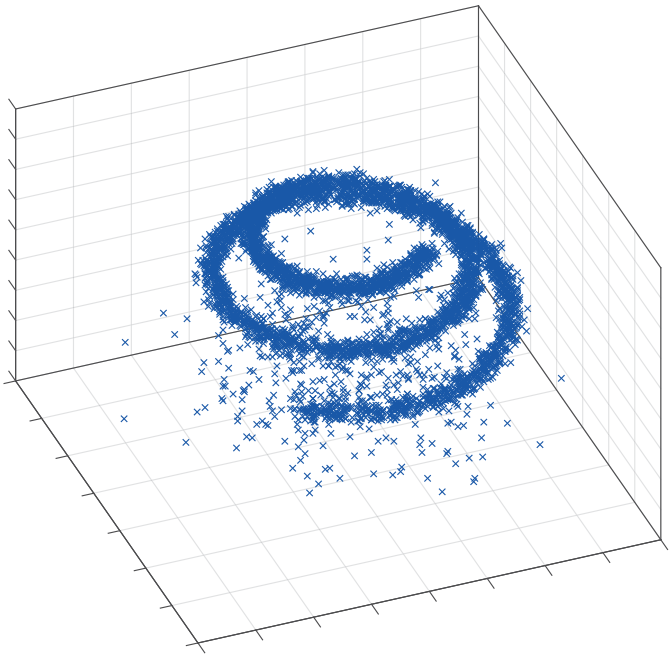


(b) 式 (5.5) の分布

図 5.2: ロバスト性の評価実験の学習データの例 (Sine 形の分布) : (a), (b) はそれぞれ 式 (5.4), 式 (5.5) の分布において 10 % のノイズを加えたデータセット.

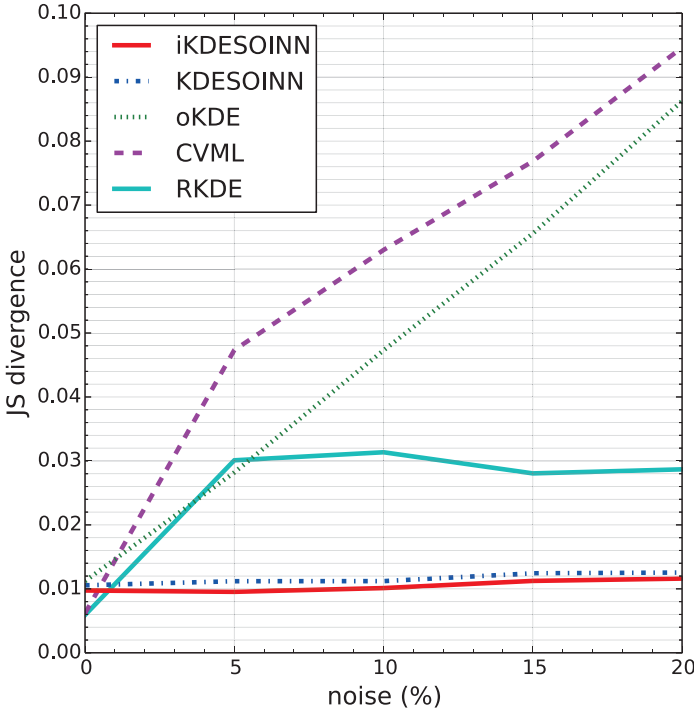


(a) 式 (5.6) の分布

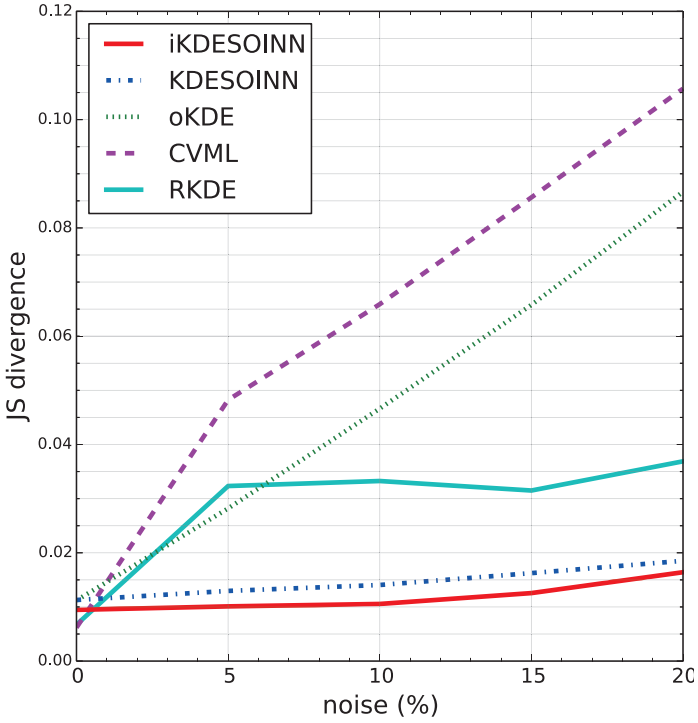


(b) 式 (5.7) の分布

図 5.3: ロバスト性の評価実験の学習データの例 (螺旋形の分布) : (a), (b) はそれぞれ 式 (5.6), 式 (5.7) の分布において 10 % のノイズを加えたデータセット.

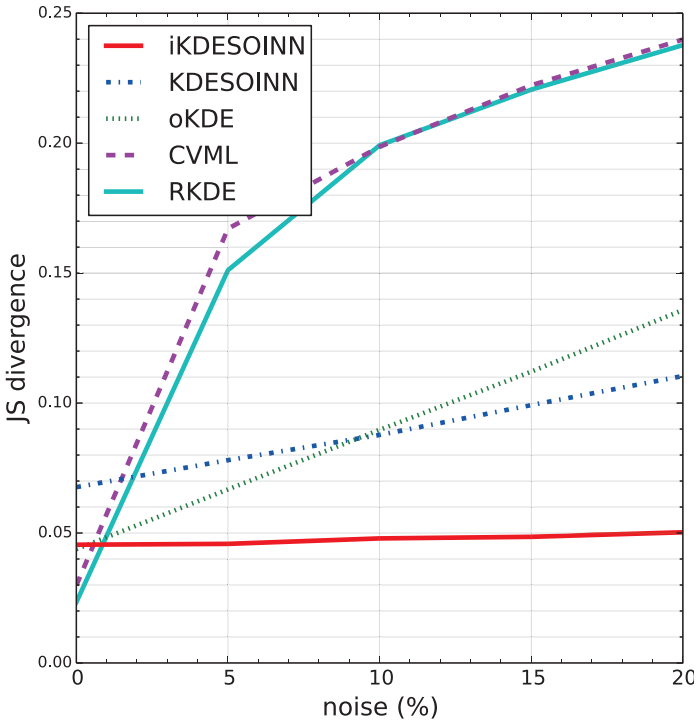


(a) 式 (5.4) の分布に対する評価実験の結果

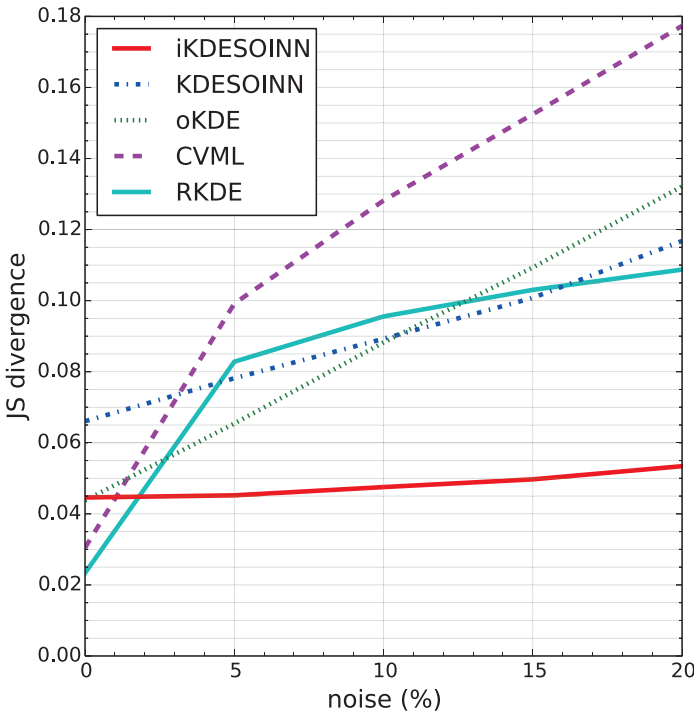


(b) 式 (5.5) の分布に対する評価実験の結果

図 5.4: ロバスト性に関する比較実験の結果 (Sine 形の分布): (a), (b) はそれぞれ 式 (5.4), 式 (5.5) の分布に対する評価実験の結果である．横軸が学習データにおけるノイズの割合 α，縦軸が推定精度の評価尺度である JS divergence を表している．



(a) 式 (5.6) の分布に対する評価実験の結果



(b) 式 (5.7) の分布に対する評価実験の結果

図 5.5: ロバスト性に関する比較実験の結果 (螺旋形の分布): (a), (b) はそれぞれ 式 (5.6), 式 (5.7) の分布に対する評価実験の結果である．横軸が学習データにおけるノイズの割合 α，縦軸が推定精度の評価尺度である JS divergence を表している．

図 5.4 及び 図 5.5 が実験結果である．図 5.4a, 図 5.4b, 図 5.5a, 図 5.5b はそれぞれ 式 (5.4), 式 (5.5), 式 (5.6), 式 (5.7) の分布に対する評価実験の結果である．横軸が学習データにおけるノイズの割合，縦軸が推定精度の評価尺度である JS divergence を表しており，JS divergence は低いほど推定精度が良い．図 5.4a 及び 図 5.5b を比較するとどの手法も同様の傾向を示しており，ノイズの分布の種類の影響は無いことが分かる．改良していない提案手法 KDESOINN も改良を加えた提案手法 iKDESOINN も高いロバスト性を示しており，図 5.4b では iKDESOINN の方が若干高い精度を示している．図 5.5a 及び 図 5.5b では他手法と比べて iKDESOINN のみが学習データにおけるノイズが増加しても高い精度を保っている．ノイズの割合が 0% の場合，既存のバッチ手法である CVML, RKDE は提案手法より高い精度を示している．しかし，ノイズの割合が 5% に増えるだけで大きく精度が低下しており，ノイズに敏感であることが分かる．実環境ではノイズを全く含まない場合は考えられないので，提案手法の方が実用的であると言える．ロバスト性のある既存手法である RKDE は図 5.5a ではロバスト性を示しているが，図 5.5b ではロバスト性のない既存手法である CVML と同程度の精度となっている．このことから RKDE のロバスト性はノイズの分布に依存することが分かる．これに対して， iKDESOINN はいずれの分布においても高いロバスト性を示しており，分布に依存していない．KDESIONN は学習データにおけるノイズの割合が 0% の場合においても他手法より精度が劣っており，ノイズの割合の増加に比例して精度が低下している．図 5.5b ではノイズの割合が 20% の時点で，ロバスト性のない既存手法の CVML と oKDE には優るものの，RKDE より劣っている．

以上のことから，改良がロバスト性向上に寄与していることが分かる．

5.4.2　学習時間

学習サンプル数に対する学習時間の変化を評価した．ノイズを含まない 式 (5.4) の分布からのサンプルを用いて，データ数を 1,000 から 100,000 まで増加させたときの計算時間の変化を既存手法と比較した．各手法のパラメータは 第 5.4.1 章と同じとした．

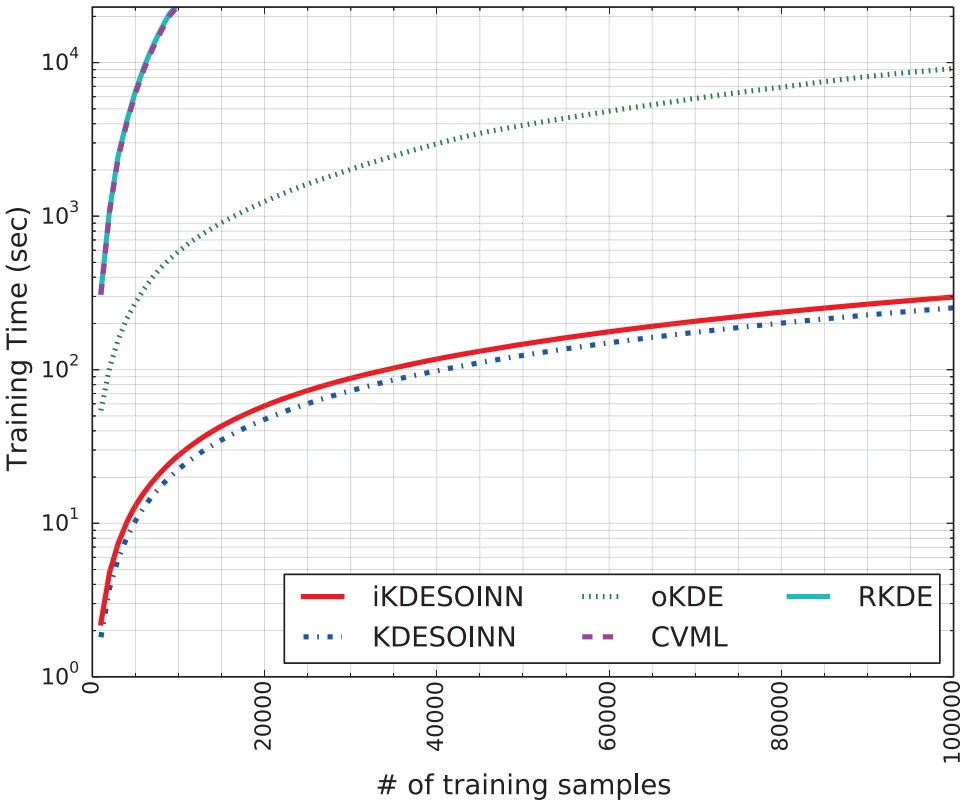


図 5.6: 学習時間に関する評価実験の結果：学習サンプル数が 1,000 から 100,000 まで変化させた時の学習時間の変化を示している．縦軸が対数目盛であることに注意されたい．

実験結果を図 5.6 に示す．縦軸が対数目盛であることに注意されたい．iKDESOINN は KDESOINN とほぼ同程度の学習速度を示している．iKDESOINN は

ネットワークが更新されると共分散行列の逆行列を再計算する必要があり，その分遅くなっていると考えられるが，更新は部分的であり学習時間にそれ程影響していない．

5.4.3　オンライン学習

オンライン学習手法はデータを追加的に学習できる．そこで，オンライン学習手法である提案手法と oKDE に対して，学習サンプルを追加した時の性能の変化を評価した．学習サンプル数を 100,000 まで追加的に増やし，1,000 サンプル学習するごとに各手法の学習時間と推定精度を評価した．サンプルを発生させる分布及び各手法のパラメータは 第 5.4.2 章 と同じ設定にした．推定精度の評価指標としては JS divergence を用いた．実験結果を図 5.7 と図 5.8 に示す．

図 5.7 は学習したサンプル数に伴う学習時間の増加量，つまり 1,000 サンプルを追加的に学習するのに要した時間の変化を評価した実験結果である．学習したサンプル数の増加に伴い， oKDE の学習時間の増加量は増加する傾向にある．これに対して，提案手法の KDESOINN と iKDESOINN の増加量は初期では増加するが，その後ほぼ一定である．つまり，提案手法の増加量はこれまで学習したサンプル数に直接的には依存しないことを示している．提案手法の学習における計算はノードとの距離計算が大きな部分を占めている．ネットワークがデータの分布を十分近似するとノードの数はほとんど変化しないため，学習時間の増加量もほとんど変化しないと考えられる．

このことから，データが継続的に生成され，データ量が増大して大規模になっていくような環境においても，提案手法は高速に学習を行えることが分かる．

図 5.8 は学習したサンプル数に伴う推定精度の変化を評価した実験結果である．oKDE は学習サンプル数が増えるに従って学習時間の増加量が大きくなる

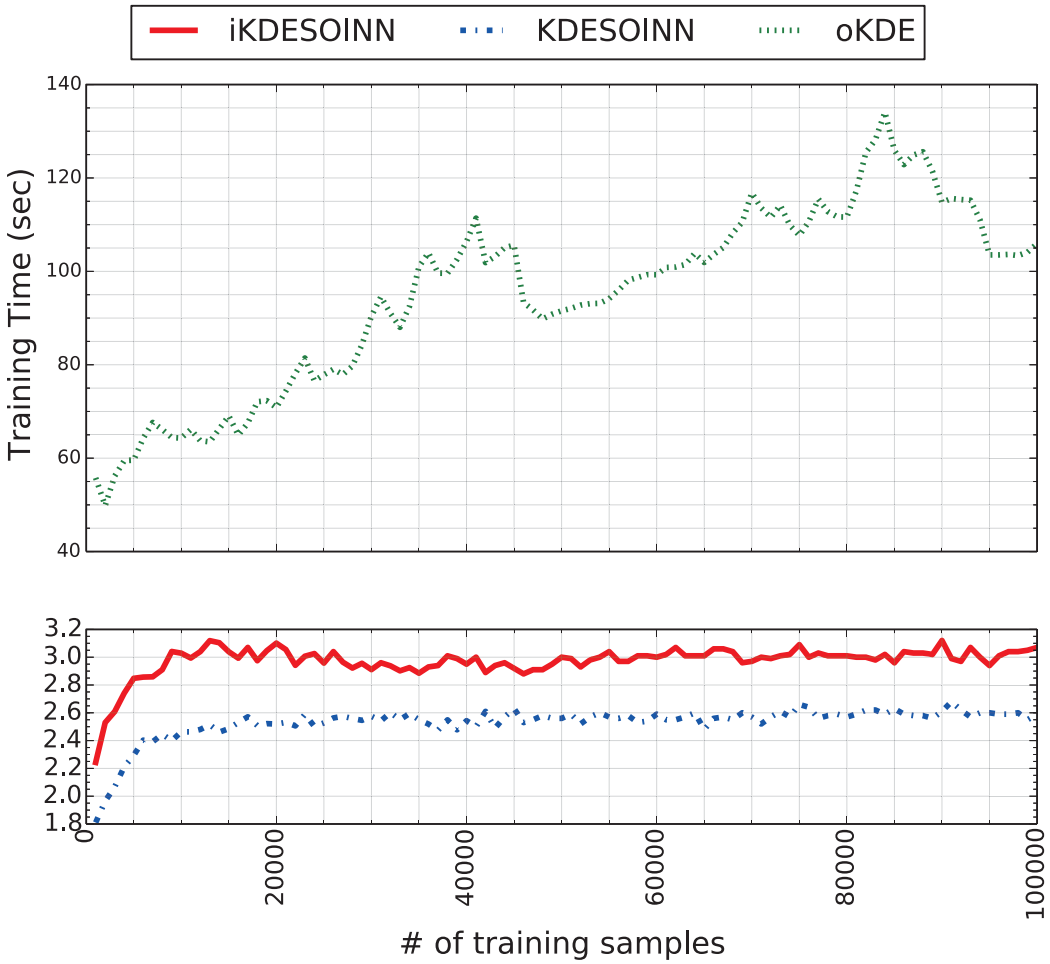


図 5.7: オンライン学習における学習時間の増加量に関する評価実験結果.

にもかかわらず，JS divergence の値は学習サンプル数が 10,000 を超えたあたりからほとんど変わらず，学習の効果が無い．KDESOINN も oKDE と同様に殆ど変化していない．これに対して，iKDESOINN はサンプル数が増えても学習時間の増加量はほとんど増えないが JS divergence の値は徐々に下がっており，精度の改善が見られる．

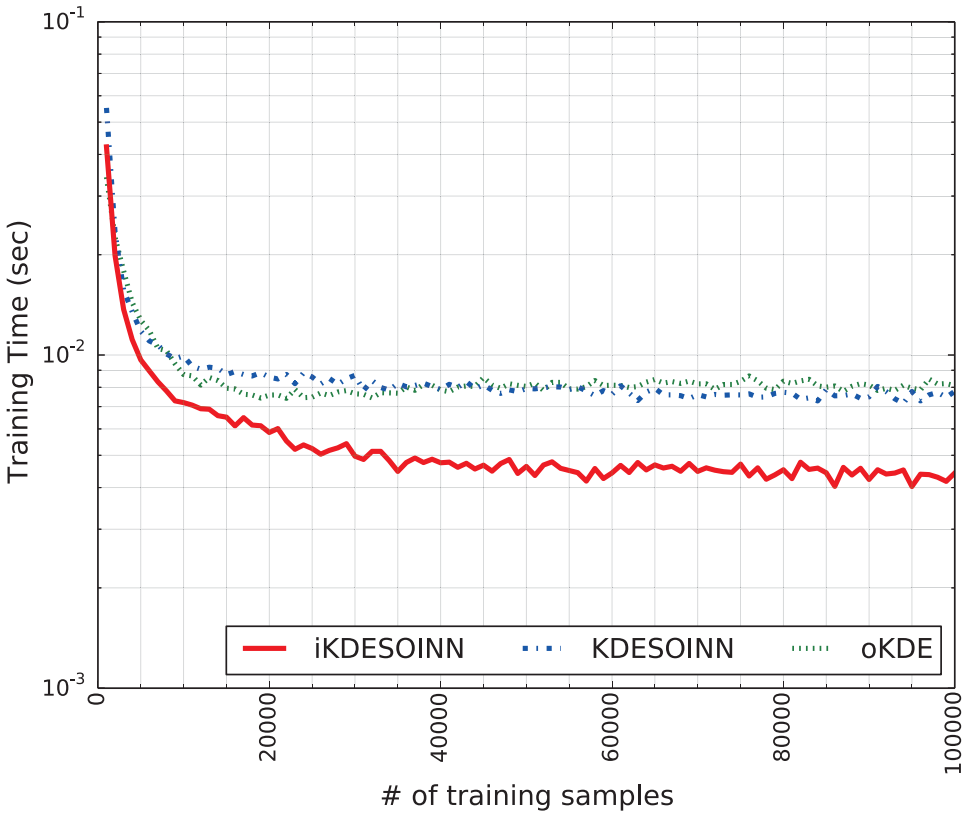


図 5.8: 学習サンプル数の増加に伴う推定精度の変化の評価実験結果.

5.4.4　高次元データ

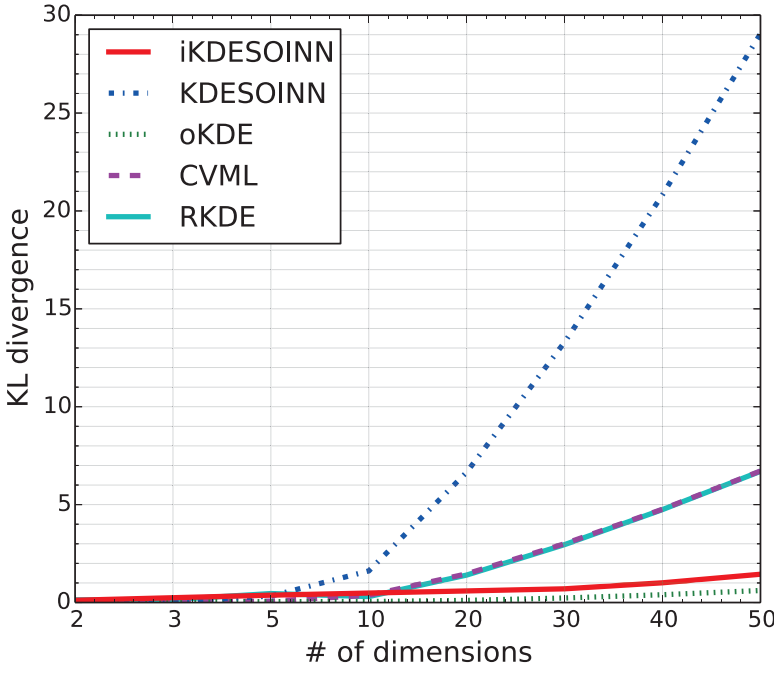
高次元データの学習に関して改良 KDESOINN と他手法を比較するために，データの次元数の増加に対する推定精度の変化を評価した．次の 2 つの分布を真の分布として用いた．

$$f(\boldsymbol{x}) \quad = \quad N(\boldsymbol{x}; \mathbf{0}, \boldsymbol{I}), \tag{5.8}$$

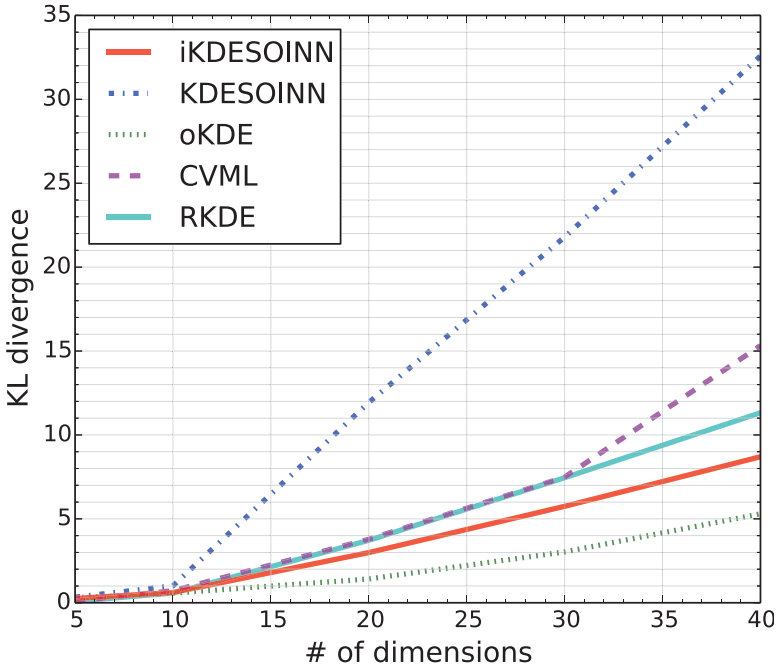
$$
\begin{aligned}
f(\boldsymbol{x}) &= N(\boldsymbol{x}; \boldsymbol{t}, 0.25 \cdot \boldsymbol{I}), \\
\boldsymbol{t} &= [a, \boldsymbol{0}]^T, a \in [-10, 10],
\end{aligned} \tag{5.9}
$$

各手法はこの分布からの 5,000 サンプルを学習し，密度推定を行う．各次元に対して 20 回ずつ実験を行い，KL divergence の平均によって各手法を評価した．式 (5.8) の分布の場合，oKDE のハイパーパラメータを $D_{th} = 10.0$，KDESOINN のハイパーパラメータを $\lambda = 2000, age_{max} = 100, \rho = 1.0, k = 2d$，iKDESOINN のハイパーパラメータを $\lambda = 2000, age_{max} = 100, \rho = 10.0, k = 2d$，とした．式 (5.9) の分布の場合，oKDE のハイパーパラメータを $D_{th} = 0.01$，KDESOINN のハイパーパラメータを $\lambda = 2000, age_{max} = 100, \rho = 1.0, k = 2d$，iKDESOINN のハイパーパラメータを $\lambda = 2000, age_{max} = 100, \rho = 1.0, k = 2d$，とした．

図 5.9 が実験結果である．横軸がデータの次元数、縦軸が推定精度の尺度である KL divergence の値を示す．KDESOINN は高次元において精度が著しく低下している．iKDESOINN は高次元でも精度が良く， CVML より高い精度である．KDESOINN は高次元空間においてはネットワークを生成できないのに対し，iKDESOINN は，閾値領域が適切に設定されると共にネットワークが調整されているため，高次元でサンプル数が限られていてもネットワークを形成できると考えられる．このため，iKDESOINN は CVML より精度が良くなっていると考えられる．oKDE が高次元においても高い精度を保っているのは，恐らく，oKDE の 圧縮スキームが単純な分布においてはとても上手く作用するためであると考えられる．



(a) 式 (5.8) の分布に対する結果



(b) 式 (5.9) の分布に対する結果

図 5.9: 高次元データに対する評価実験の結果：横軸がデータの次元数、縦軸が推定精度の尺度である KL divergence の値を示す.

5.4.5　実データによる精度評価

第 4.4.4 章 と同様に UCI Machine Learning Repository のデータセットを使って実データにおける確率密度推定を評価した．実験に用いたデータセットを 表 5.1 に示す．第 4.4.4 章 と比べてより高次元のデータセット，よりサンプル数の多いデータセットを追加した．データセットのクラスごとに実験を行った．ただし，Skin は正例，負例の 2 クラスあるが，正例 (50,859 サンプル) のみ用いた．各実験のサンプル集合を白色化により標準化し，そこからランダムに抽出した 75 %のサンプルを学習データ，残りをテストデータとし，密度関数の推定精度を評価した．これを各実験で 20 回繰り返し，データセットごとに平均し評価した．評価尺度には Negative Log Likelihood (NLL) を用いた．NLL は無限大に発散する可能性があるため，順序平均をとることで対応した．oKDE のハイパーパラメータを $D_{th} = 0.1$，KDESOINN のハイパーパラメータを $\lambda = 300, age_{max} = 50, \rho = 0.5$，iKDESOINN のハイパーパラメータを $\lambda = 1000, age_{max} = 100, \rho = 0.6, k = 2d$ とした．

表 5.1: 第 5.4.5 章 における精度評価で用いた実データセット一覧

	#samples	#dimension	#classes
Iris	150	4	3
Pima	768	8	2
Wine	178	13	3
Wine_Red	1,599	11	6
Wine_White	4,898	11	7
MAGIC	19,020	10	2
Skin	50,859	3	1
SUSY	2,287,827	18	1
HIGGS	5,829,123	28	1

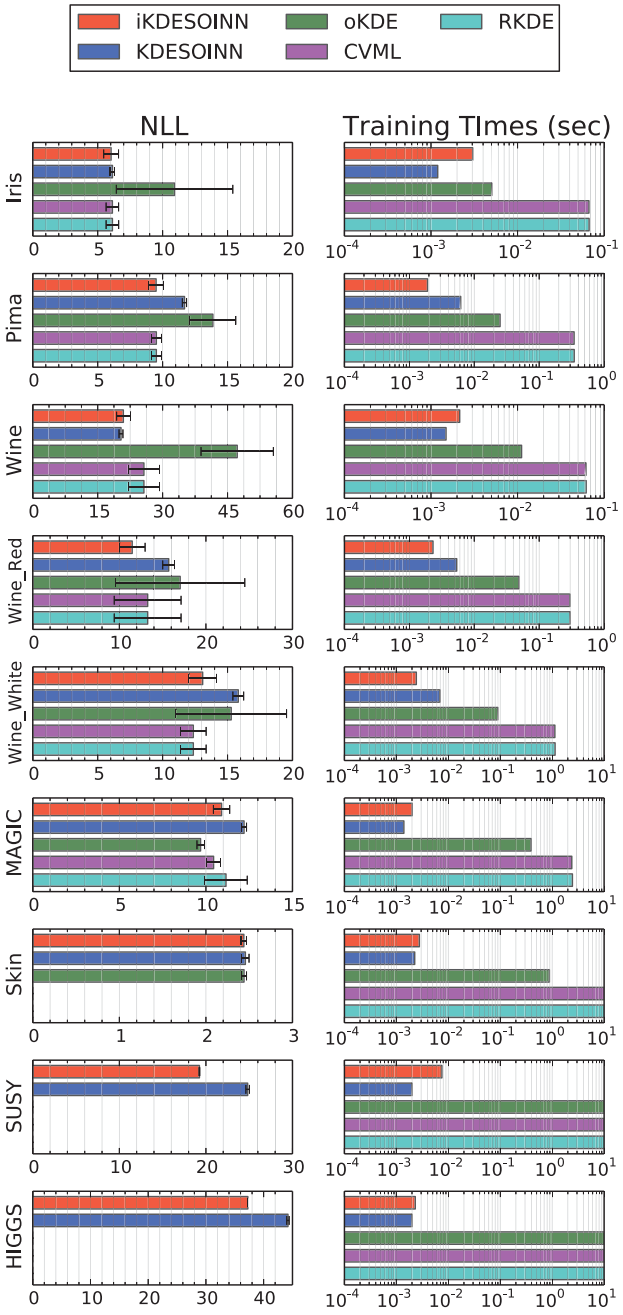


図 5.10: 実データに対する密度推定の評価実験の結果： 各行が各データセットに対する実験結果である．左の列が推定精度である Negative Log Likelihood (NLL) の値を示しており，棒グラフが平均，エラーバーが標準偏差を表している．右の列が 1 サンプルあたりの学習時間を示している．こちらのグラフは対数目盛であることに注意されたい．

実験結果を 図 5.10 に示す．各行が各データセットに対する実験結果である．左の列が推定精度である NLL の値を示しており，棒グラフが平均，エラーバーが標準偏差を表している．右の列が 1 サンプルあたりの学習時間を示している．こちらのグラフは対数目盛であることに注意されたい．Skin, SUSY, 及び HIGGS データセットにおいて，幾つかの手法は 5 日経過しても 1 度も実験が終わらなかったため，NLL の値を示していない．Iris, Wine, 及び Skin データセットにおいては，iKDESOINN は KDESOINN と同程度の精度であるが，それ以外のデータセットにおいては大幅に精度が向上している．これは，iKDESOINN は，閾値領域が適切に設定されると共にネットワークが調整されているためであると考えられる．oKDE は，第 5.4.4 章 の実験においては高い精度だったにも関わらず，精度が良くない．これは実データの分布が複雑であるためである考えられる．2 つの提案手法以外の手法は，学習データの量が増えるに従って，学習時間が増加する傾向にある．KDESOINN は iKDESOINN より僅かに高速だが，精度が低い．特に高次元データで顕著である．以上のことから，iKDESOINN は実データに対する密度推定に関しても実用的に適用できると言える．

5.5　考察

5.5.1　アルゴリズムの改良の効果の検証

ここでは，提案手法における学習アルゴリズムの改良が，確率密度推定の性能向上に如何に寄与しているかを，SOINN と比較することで検証したい．

第 5.4.1 章の式 (5.4) に対する実験と同様の実験を行う．学習アルゴリズムに改良を加えた提案手法 iKDESOINN と，学習アルゴリズムのみを SOINN にし，他の部分は提案手法と同一にした手法を比較する．各種法において学習ア

ルゴリズムのハイパーパラメータは $\lambda = 500, age_{max} = 100, \rho = 0.1, k = 2d$ とした．

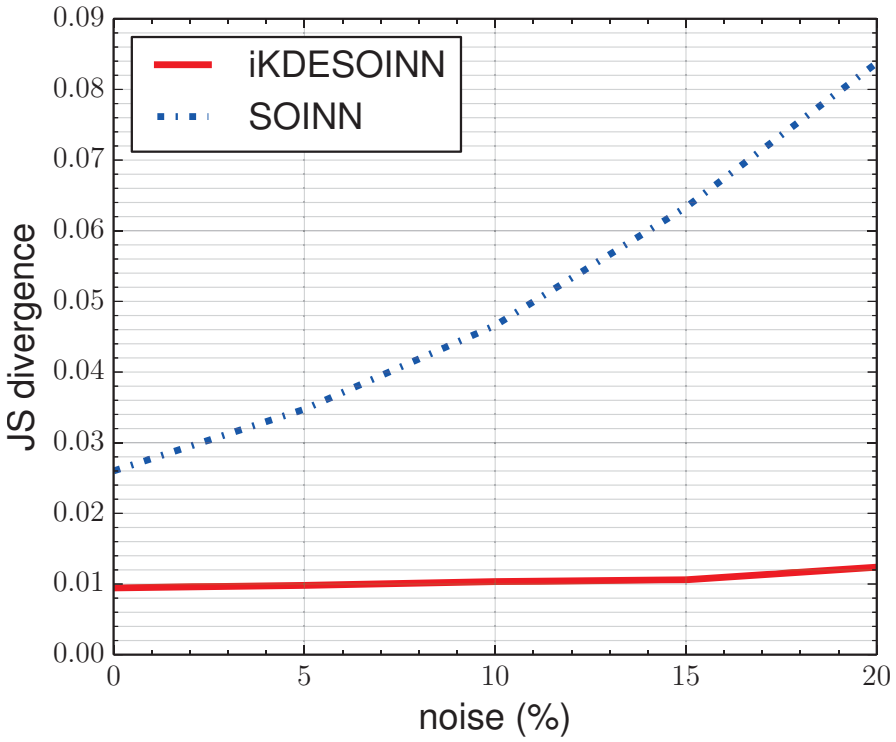


図 5.11: ロバスト性に関する学習アルゴリズムの改良の検証実験の結果．

図 5.11 が実験の結果である．iKDESOINN の方が全体を通して精度が高く，特にノイズの割合が増加すると顕著になる．従来の SOINN の閾値領域は超球であり，ネットワークに沿うものではない．そのため，分布を適切に表現しないエッジができてしまい，分布の推定精度が低下してしまうとともに，ノイズの影響を受けやすくなってしまう．エッジ作成時の閾値領域を改良している iKDESOINN の学習アルゴリズムが，サンプル集合の密度関数を表すネットワーク構造を SOINN よりも学習できていると考えられ，実験結果からも示されている．

5.5.2　ハイパーパラメータ

ここでは，改良 KDESOINN おけるハイパーパラメータが性能に与える影響に関して考察する．第 5.4.1 章 と同様の実験をハイパーパラメータを変えて行い，精度の変化を評価した．分布には 式 (5.4) 及び 式 (5.7) を用いた．第 5.4.1 章 で用いた各ハイパーパラメータの値を基準とし，対象のハイパーパラメータのみを変化させ，その他のハイパーパラメータは基準値とした．

図 5.12 及び図 5.13，図 5.14 が評価結果である．横軸が対象のハイパーパラメータの値を表す．対数スケールであることに注意されたい．縦軸が推定精度の評価尺度である JS divergence を表しており， JS divergence は低いほど推定精度が良い．各折れ線グラフは学習データに含まれるノイズの割合が同一であるものを表している．

図 5.12 ではノード削除ハイパーパラメータ λ の変化による精度の変化を評価している．λ は定期的に行うノイズノード削除のインターバルの間に入力される入力サンプル数を表している．いずれの分布においても λ の値が小さいと精度が大きく低下しているのが分かる．λ が小さいと，ノイズノード削除が多く行われることになる．新規ノードは，エッジが張られネットワークに取り込まれるまではノイズノードとして認識されているため，ノイズノードの削除が頻繁に行われると，新規ノードがネットワークに取り込まれず，ネットワークが十分に成長しないと考えられる．そのため，λ が小さくなると精度が大きく低下すると考えられる．また，λ の値が大きくなりすぎるとロバスト性が低下する傾向にある．学習データがノイズを含まない場合，λ が大きくなっても精度は低下しないのに対し，学習データにおけるノイズの割合が大きいと，精度が低下している．λ の値が大きいとノイズノード削除の頻度が小さくなり，ノ

イズとして削除されるべきノードまでがネットワークに取り込まれてしまうためだと考えられる．

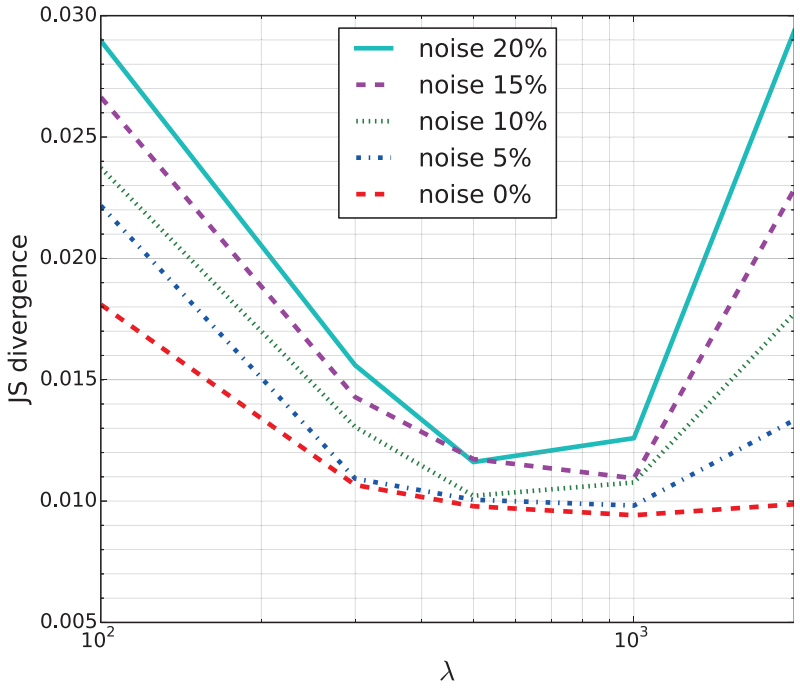
図 5.12 の結果から，λ は数百以上，学習データにノイズが含まれる場合は値を大きくし過ぎないことが推奨される．また，提案手法の学習アルゴリズムの計算量は主にノードとの距離計算が占めている．λ の値が大きいと，新規ノードが削除されにくくなり，ノード数が増える傾向にある．そのため，計算量を考慮する場合は λ はなるべく小さい値にすべきである．

図 5.13 ではエッジ削除ハイパーパラメータ age_{max} の変化による精度の変化を評価している．age_{max} は， adge aging scheme によるエッジの削除の閾値となる．エッジは更新が行われないと，年齢が増加する．つまり，年齢が大きなエッジは更新が行われておらず，不適切であることを意味している．age_{max} はそのような不適切なエッジを削除するための基準であり，エッジの年齢の最大値を表す．age_{max} は他のハイパーパラメータと比較すると，性能に大きな影響を与えていない．改良 KDESOINN では閾値領域を改良しているため不適切なエッジが生成されづらく，性能に影響を与えにくいのではないかと考えられる．また，学習サンプル数が少ない場合，各エッジの年齢が age_{max} を超えることはあまり起こらないため，edge aging scheme によるエッジの削除があまり行われず，age_{max} の影響が小さいのではないかと考えられる．しかし，学習サンプル数が膨大になった際は，edge aging scheme によるエッジの削除によってネットワークの不要な膨張を抑える効果があると考えられる．したがって，age_{max} はなるべく小さく設定すべきである．

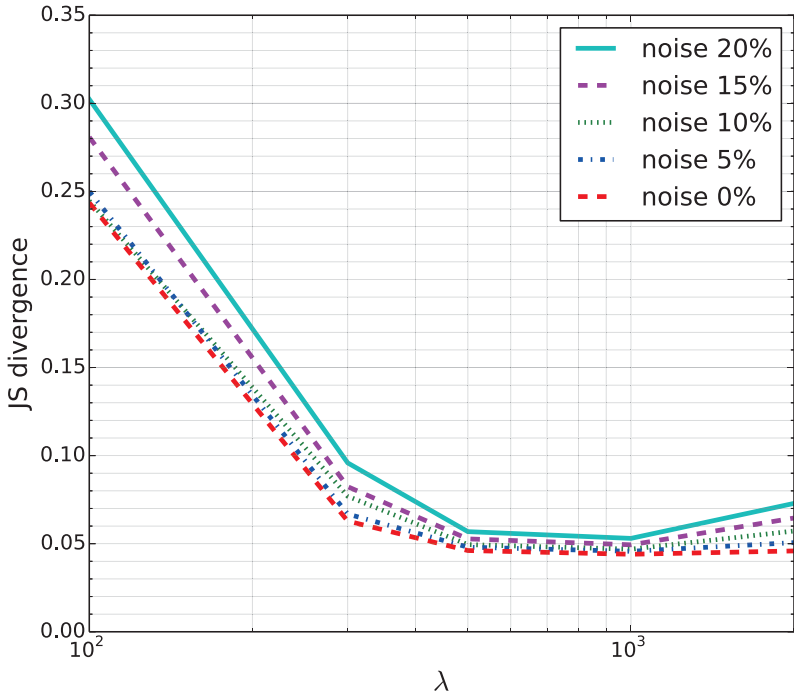
図 5.14 では閾値領域補正ハイパーパラメータ ρ の変化による精度の変化を評価している．ρ の値によって性能が大きく変化しており最も性能に影響を与えていることが分かる．いずれの分布でも ρ の値が小さすぎる又は大きすぎる

場合に大きく精度が低下している．また，それぞれで最適な値が異なっている．式 (5.7) では $\rho < 10^{-1}$ の場合には，全くネットワークが構成されない場合もあった．式 (5.3) から分かるように ρ はネットワークに接続していないノードの閾値領域の大きさを決定している．ρ が小さすぎると，ネットワークに接続していないノードがネットワークに取り込まれづらく，ノイズとして全て削除されてしまい，全くネットワークが構成されないということが発生してしまうと考えられる．ρ が大きくなると，閾値領域が大きくなり，閾値領域による制約がない場合，つまりオリジナルの Competitive Hebbian Learning に近づくと考えられる．式 (5.4) の分布で学習データにおけるノイズの割合が 0 % の場合，$\rho > 2$ で精度がほぼ横這いとなっている．これは，閾値領域が大きくなり制約として機能しなくなったため，ρ を大きくしても意味がなく，Competitive Hebbian Learning と同様のネットワークを構成するようになったためだと考えられる．

図 5.14 の結果から，ρ は適切に最適化する必要があると言える．経験的には，ρ の最適値は，次元数，スケール，データの分布の複雑さに影響を受けると考えられる．そのため，前処理として標準化を行うことである程度 ρ の影響を抑えることができると考えられる．

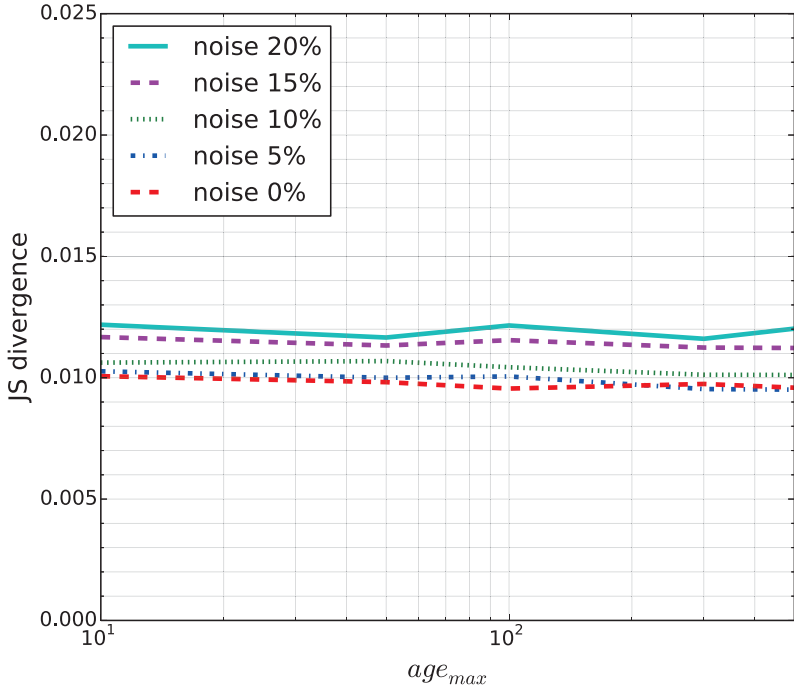


(a)

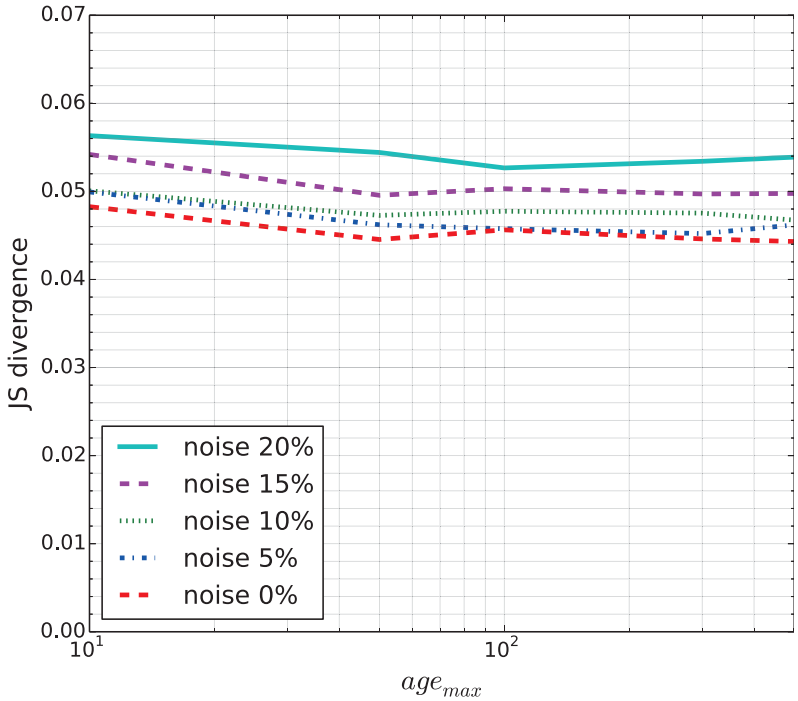


(b)

図 5.12: ノード削除ハイパーパラメータ λ の変化による性能の変化:

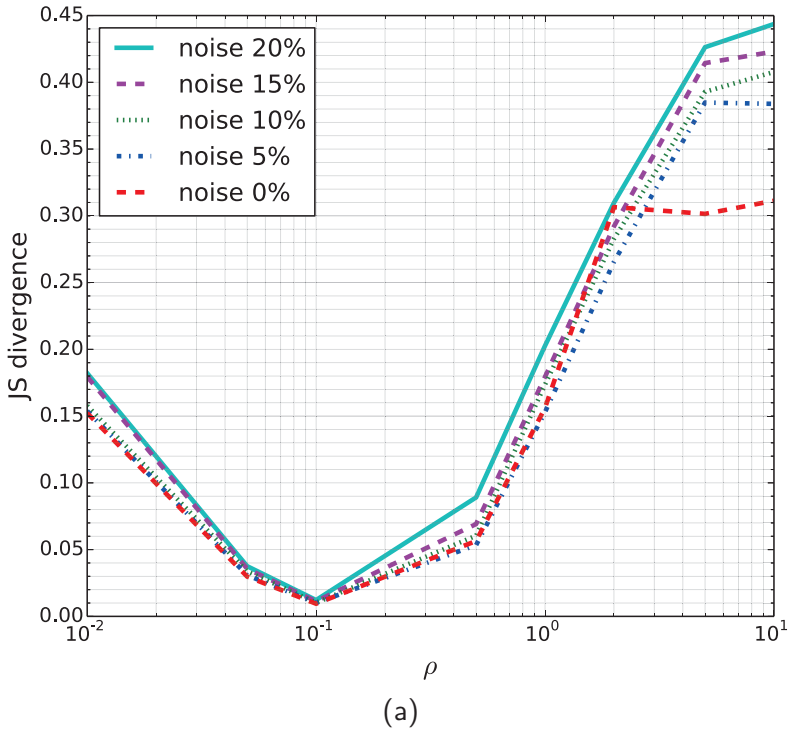


(a)



(b)

図 5.13: エッジ削除ハイパーパラメータ age_{max} の変化による性能の変化:



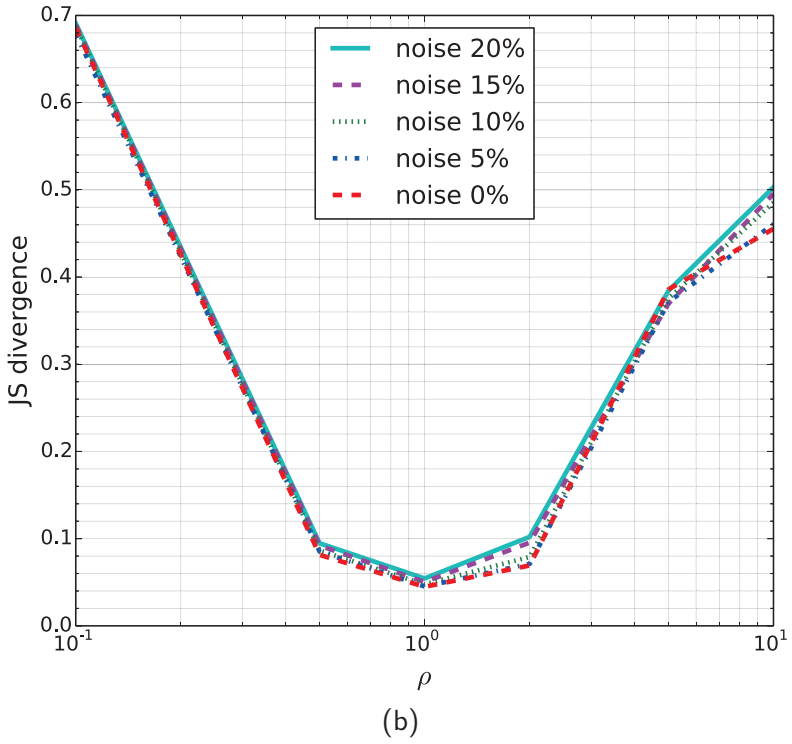


図 5.14: 閾値領域補正ハイパーパラメータ ρ の変化による性能の変化:

6 結論

本研究では，自己増殖型ニューラルネットワーク (Self-Organizing Incremental Neural Network: SOINN) とカーネル密度推定法 (Kernel Density Estimation: KDE) を拡張し，ロバスト性を有し，高速にオンライン学習が可能なノンパラメトリック確率密度推定法を考案した．提案手法は， IoT などデータストリームから生成される大規模データ活用するための知的情報処理の基盤技術となる手法であると考えている．ネットワークやセンサー技術の発展に伴って，実環境から様々なデータが膨大に高速に継続的に生成されるようになったのが，データ量の劇的な増加の一因である．実環境から取得されるノイズの多いデータに対応するために高いロバスト性が，高速に継続的に生成されるデータに対応するために高速なオンライン学習が，様々なデータの解析に対応するためにノンパラメトリックな手法が必要であり，提案手法は，これら 3 つの特徴を全て有している．

第 3 章 では，提案手法の基盤となっている SOINN について，確率密度推定に応用できる統計的な意味付けに関して述べた．SOINN では，競合学習によるベクトル量子化によってサンプル集合からその代表であるプロトタイプの集合

を算出する．各プロトタイプは競合学習において勝ち得たサンプルの平均に位置し，競合学習の勝者になった回数が周辺の密度を表している．さらに SOINN では，Competitive Hebbina Learning によってこれらのプロトタイプをノードとするネットワークを形成する．このネットワークはサンプルの分布に沿うように形成される．

第 4 章 では，前章で述べた SOINN のネットワークの統計的な性質を利用し，KDE を拡張したノンパラメトリック密度推定法を提案した．ネットワークの各ノードの位置にガウスカーネルを配置し，それらの和として推定密度関数を定義する．ガウスカーネルの共分散行列はネットワークの性質が反映されるように設計されており，各ノード周辺のネットワークの形状によって適応的に決定される．評価実験において，ロバスト性と学習時間に関して，提案手法は既存手法より高い性能を示した．

第 5 章 では，学習アルゴリズムを改良することで，より密度推定に適したネットワーク構造を形成するようにし，性能向上を試みた．SOINN において最も重要な部分である閾値領域に関して考察し，より不要なエッジが形成されないように改良した．評価実験によって，ロバスト性，高次元データに対する密度推定，実データに対する推定精度に関して大きく性能が向上していることを確認した．

また，第 5.4.3 章 の評価実験の結果より，学習が十分であれば，提案手法の計算量がサンプルデータ数に依存しないことが分かり，継続的に無限に生成されるデータに対しても対応できることが示された．ノンパラメトリック密度推定の代表的な手法である KDE は精度を高めるために大量のサンプル数が必要であり，サンプル数の増加に伴って計算量が大きくなってしまう．これに対して，提案手法はサンプル数よりも大幅に少ないプロトタイプを用い，更にそれ

らにネットワークという構造を加えることで計算量を大きく抑えていると共に，十分に学習すれば，プロトタイプの数は増加しないため，計算量の増加を抑えることができている．

6.1　課題

提案手法の課題としては，高次元への対応，ハイパーパラメータの影響の低減，理論的な解析が挙げられる．

6.1.1　高次元への対応

一般にノンパラメトリック密度推定法は高次元データに対しては推定精度が良くない [54,55]．提案手法もノンパラメトリック密度推定法であるためデータの次元数が大きくなるに連れて推定精度が低下してしまうことが，第 5.4.4 章によって示されている．これに対処するために，次元削減や特徴抽出の手法と統合する事が考えられる．オンラインに主成分分析およびカーネル主成分分析を行う手法 [56–58] が考案されているため，これらと組み合わせることで，提案手法のオンライン性を失うことなく高次元データを低次元に写像し密度推定法することができる．

6.1.2　ハイパーパラメータの影響の低減

提案手法の学習アルゴリズムはオリジナルの SOINN よりハイパーパラメータの数が増加してしまっており，それらの決定にはより手間が掛かる．現状，実験においては， Cross-Validation を用いて Grid Search を行うことで全てのハ

イパーパラメータの決定を行っている．提案手法の学習は高速であるが，全てのハイパーパラメータを決定するには手間が掛かる．

第 5.5.2 章 ではハイパーパラメータの性能への影響を考察しており，幾つかのハイパーパラメータは適切に設定しないと性能が大きく低下することを示している．ノード削除ハイパーパラメータ λ はロバスト性と推定精度に影響を与えており，大きすぎるとロバスト性が低下し，小さすぎると推定精度が低下してしまう．閾値領域補正ハイパーパラメータ ρ は推定精度に大きく影響を与えており，データによって適切な値が異なる．

ハイパーパラメータの影響を低減させるために，センシティブなハイパーパラメータの廃止，またはハイパーパラメータを自動決定する機構を導入することが必要である．

6.1.3　理論的な解析

本研究では，SOINN が形成するネットワークが分布の形状を学習できていることを示し，それを応用して確率密度推定を行った．競合学習によって作成されたサンプルのプロトタイプと Competitive Hebbian Learning によってプロトタイプ間に形成されたネットワークが分布の形状を表していることを示した．この性質を引き出すように推定密度関数を設計し，その有効性を評価実験によって実験的に証明した．

しかし，形成するネットワークが，提案手法の推定密度関数を用いた密度推定に最適なのかどうかは理論的には証明できていない．これは提案手法が基礎としている SOINN が理論的に解析されていないことに起因する．第 3.2.2 章において，SOINN における競合学習によるベクトル量子化が k-Means と同様の最適化を行っていることを指摘したが，アルゴリズム全体でどのような目的

関数を最適化しているのかは示されていない．密度推定の観点から目的関数を設定できれば，提案手法の理論的な裏付けをより強固にできる．

6.2　今後の展望

ノイズを含む実環境からのストリームデータの分布を推定できるようになれば，確率的な予測や推論がより実際的なアプリケーションとして構築でき，その応用範囲は広いと考えられる．ここで，いくつか例を挙げる．

6.2.1　データストリームに対する異常検知

データストリームにおいても異常検知は重要なタスクである．例えば，ウェアラブルセンサーによって心拍をモニターしている状態では，センサーの異常値は心臓発作などの前兆である可能性があり，検出する必要がある．

提案手法は高いロバスト性を有するため，データストリームに対する異常検知に有効であると考えられる．提案手法は異常値を含むデータからでも通常時のデータの分布を推定することができるため，前もって異常値と通常時のデータを分ける必要がない．このため，データストリームのような高速に継続的にデータが生成されるような環境においても，異常検知システムの構築が可能である．

また，提案手法の学習アルゴリズムでは，ノイズを削除するプロセス (アルゴリズム 4 の 19, 22 行目) があり，どのサンプルがノイズだったかを判断することもできる．

6.2.2　データストリームに対する確率的な予測

提案手法はデータストリームを用いた知的情報処理に応用することができると考えられる．

例えば，車に取り付けられた外界センサーの情報から，車がどのように進むか予測し，事故を起こさないようにドライバーを補助するシステムなどが考えられる．車に取り付けられたセンサーは，様々な環境下でセンシングすることが求められる．このため，提案手法のようなロバストな手法が必要である．また，ドライバーの癖やセンサーの種類，車種によって得られるデータが異なってくるので，ノンパラメトリックな手法が有効であり，提案手法が適している．

このような知的情報処理は，確率的な予測によって実現することができる．入力データ $\boldsymbol{x}$ から y を予測する場合を考える．クラス分類問題など y が離散値 $y_1 \cdots y_N$ をとる場合は，

$$y = \underset{y_i}{\operatorname{argmax}} \ p(y_i|\boldsymbol{x})$$

と定式化できる．ベイズの定理より

$$p(y_i|\boldsymbol{x}) \propto p(\boldsymbol{x}|y_i)p(y_i)$$

であり，$p(y_i)$ は y_i に対応する学習データをカウントするとこで容易に求まるので，y_i に対応する入力データを用いて $p(\boldsymbol{x}|y_i)$ を，提案手法を用いて，それぞれ求めることで，式 (6.1) を解くことができる．

$\boldsymbol{y}$ が連続値をとる場合，条件付き確率 $p(\boldsymbol{y}|\boldsymbol{x})$ を直接求める必要がある．$\boldsymbol{y}$ が

連続値をとる例としては回帰問題が考えられる．回帰関数 $\boldsymbol{y}(\boldsymbol{x})$ は $p(\boldsymbol{y}|\boldsymbol{x})$ から

$$\boldsymbol{y}(\boldsymbol{x}) = E[\boldsymbol{y}|\boldsymbol{x}] = \int \boldsymbol{y} p(\boldsymbol{y}|\boldsymbol{x}) dy$$

と定式化できる．

提案手法を用いて条件付き確率を求めるには，カーネル密度推定を用いて条件付き確率を求める手法である Nadaraya-Watson モデル [59,60] を用いる方法が考えられる．また，提案手法の推定密度関数は，ガウス分布をカーネル関数に用いていおり，ガウス分布の混合モデルとしてみることもできるので，ガウス混合モデルに対する条件付き確率を求める手法 [61] を用いることもできると考えられる．

6.2.3　データストリーム間のベイジアンネットワーク

より知的な情報処理を行うために，複数のデータストリーム間にベイジアンネットワークを構築し，推論を行うことも考えられる．

例えば，複数のウェアラブルセンサーから生成される生体データのデータストリームから，ユーザーの現在の行動や，健康状態，感情などの推論をリアルタイムで行い，それに応じた提案を行うアプリケーションなどが考えられる．ウェアラブルセンサーは人に取り付けられるため，様々な環境下でセンシングすることが求められる．このため，提案手法のようなロバストな手法が必要である．また，生体データは個人差が大きいため，個々人に適応する必要があり，ノンパラメトリックな手法が有効であり，提案手法が適している．

ベイジアンネットワークを構築するには，確率変数間の関係を表すネットワーク構造と確率変数間の条件付き確率を求める必要がある．確率変数間の条件付

き確率分布は 第 6.2.2 章 の方法を用いて提案手法によってノンパラメトリックに求められる．確率変数間のネットワーク構造に関してはデータから学習することも考えられている [62]．時間変化する動的ベイジアンネットワークのネットワーク構造を学習する手法 [63,64] と組み合わせることによって，時々刻々と変化するデータ間の関係も捉えることも可能になるのではないかと考えられる．

関連研究

学術論文誌

- Y. Nakamura, O. Hasegawa, ”Non-parametric Density Estimation based on Self-Organizing Incremental Neural Network for Large Noisy Data”, IEEE Trans. Neural Networks and Learning Systems, 2016.
- 中村圭宏, 長谷川修, ”ロバスト高速オンライン多変量ノンパラメトリック密度推定法” , 電子情報通信学会 和文論文誌D,Vol. J97-D, No.8, pp.1284-1296, 2014.

国際会議論文

- Y.Nakamura, O.Hasegawa, ”Robust Fast Online Multivariate Non-parametric Density Estimator”, 20th International Conference on Neural Information Processing, 2013.

国内会議／口頭発表

- 中村圭宏, 高橋大志, 長谷川修, ”ノイズ環境下にける高速オンライン多変量ノンパラメトリック密度推定法”, 電子情報通信学会 パターン認識・メディア理解研究会, 2013.
- 高橋大志, 中村圭宏, 長谷川修, ”自己増殖型ニューラルネットワークを用いた高次元ノンパラメトリック確率密度推定”, 電子情報通信学会 パターン認識・メディア理解研究会, 2013.
- 中村圭宏, 高橋大志, 長谷川修, ”耐ノイズ性を有する高速多変量オンラインノンパラメトリック密度推定法”, 第 15 回情報論的学習理論ワークショップ, 2012.

参考文献

[1] Turner, V., Gantz, J. F., Reinsel, D. and Minton, S.: The digital universe of opportunities: Rich data and the increasing value of the internet of things, International Data Corporation, White Paper, IDC_1672 (2014).

[2] Mayer-Schönberger, V. and Cukier, K.: Big Data: A Revolution that Will Transform how We Live, Work, and Think, John Murray Publishers, London (2013).

[3] Fan, W. and Bifet, A.: Mining big data: current status, and forecast to the future, SIGKDD Explor. Newsl., Vol. 14, No. 2, pp. 1–5 (2013).

[4] Laurila, J. K., Gatica-Perez, D., Aad, I., Blom, J., Bornet, O., Do, T., Dousse, O., Eberle, J. and Miettinen, M.: The mobile data challenge: Big data for mobile computing research, Pervasive Computing (2012).

[5] Howe, D., Costanzo, M., Fey, P., Gojobori, T., Hannick, L., Hide, W., Hill, D. P., Kania, R., Schaeffer, M., St Pierre, S., Twigger, S., White, O. and Rhee, S. Y.: Big data: The future of biocuration, Nature, Vol. 455, pp. 47–50 (2008).

[6] Atzori, L., Iera, A. and Morabito, G.: The Internet of Things: A survey, Computer Networks, Vol. 54, No. 15, pp. 2787 – 2805 (2010).

[7] Weiser, M.: The computer for the 21st century, Scientific american, Vol. 265, No. 3, pp. 94–104 (1991).

[8] Khaitan, S. and McCalley, J.: Design Techniques and Applications of Cyberphysical Systems: A Survey, Systems Journal, IEEE, Vol. 9, No. 2, pp. 350–365 (2015).

[9] Silva, J. A., Faria, E. R., Barros, R. C., Hruschka, E. R., Carvalho, A. C. P. L. F. d. and Gama, J. a.: Data Stream Clustering: A Survey, ACM Comput. Surv., Vol. 46, No. 1, pp. 13:1–13:31 (2013).

[10] Krempl, G., Žliobaite, I., Brzeziński, D., Hüllermeier, E., Last, M., Lemaire, V., Noack, T., Shaker, A., Sievi, S., Spiliopoulou, M. et al.: Open challenges for data stream mining research, ACM SIGKDD Explorations Newsletter, Vol. 16, No. 1, pp. 1–10 (2014).

[11] Hornik, K., Stinchcombe, M. and White, H.: Multilayer feedforward networks are universal approximators, Neural Networks, Vol. 2, No. 5, pp. 359 – 366 (1989).

[12] Seide, F., Li, G. and Yu, D.: Conversational Speech Transcription Using Context-Dependent Deep Neural Networks, Interspeech 2011, International Speech Communication Association (2011).

[13] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C.

and Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge, International Journal of Computer Vision (IJCV), Vol. 115, No. 3, pp. 211–252 (2015).

[14] : Human-level control through deep reinforcement learning, Nature, Vol. 518, No. 7540, pp. 529–533 (2015).

[15] Huber, P.: Robust statistics, Wiley, New York (1981).

[16] Hubert, M., Rousseeuw, P. J. and Vanden Branden, K.: ROBPCA: A New Approach to Robust Principal Component Analysis, Technometrics, Vol. 47, No. 1, pp. 64–79 (2005).

[17] Xanthopoulos, P., Pardalos, P. M. and Trafalis, T. B.: Robust data mining, SpringerBriefs in Optimization, Springer, New York (2013).

[18] Kim, J. and Scott, C. D.: Robust Kernel Density Estimation, Journal of Machine Learning Research, Vol. 13, pp. 2529–2565 (2012).

[19] Parzen, E.: On estimation of a probability density function and mode, The Annals of Mathematical Statistics, Vol. 33, pp. 1065–1076 (1962).

[20] Silverman, B.: Density Estimation for Statistics and Data Analysis, Chapman and Hall, London (1986).

[21] Hall, P., Sheather, S., Jones, M. and Marron, J.: On optimal data-based bandwidth selection in kernel density estimation, Biometrika, Vol. 78, No. 2, pp. 263–269 (1991).

[22] Jones, M. C., Marron, J. S. and Sheather, S. J.: A brief survey of bandwidth selection for density estimation, Journal of the American Statistical Association, Vol. 91, No. 433, pp. 401–407 (1996).

[23] Bowman, A.: An alternative method of cross-validation for the smoothing of density estimates, Biometrika, Vol. 71, pp. 353–360 (1984).

[24] Chaudhuri, D., Chaudhuri, B. and Murthy, C.: A data driven procedure for density estimation with some applications, Pattern Recognition, Vol. 29, No. 10, pp. 1719–1736 (1996).

[25] Leiva, J. M. and ArtÃľs, A.: Algorithms for Gaussian bandwidth selection in kernel density estimators, Neural Information Processing Systems, Workshop on Optimization for Machine Learning, Vancouver, B.C., Canada (2008).

[26] Declercq, A. and Piater, J.: Online learning of Gaussian mixture models — a two level approach, Proceedings of the 3rd International Conference on Computer Vision Theory and Applications, Funchal, Portugal, pp. 605–611 (2008).

[27] Tabata, K., Sato, M. and Kudo, M.: Data compression by volume prototypes for streaming data, Pattern Recognition, Vol. 43, No. 9, pp. 3162–3176 (2010).

[28] Kristan, M., Leonardis, A. and Skočaj, D.: Multivariate online kernel density estimation with Gaussian kernels, Pattern Recognition, Vol. 44, No. 10—11, pp. 2630–2642 (2011).

[29] Shen, F. and Hasegawa, O.: An incremental network for on-line unsupervised classification and topology learning, Neural Networks, Vol. 19, No. 1, pp. 90–106 (2006).

[30] Shen, F. and Hasegawa, O.: A Fast Nearest Neighbor Classifier Based on Self-organizing Incremental Neural Network, Neural Networks, Vol. 21, No. 10, pp. 1537–1547 (2008).

[31] Leiva-Murillo, J. M. and ArtéS-RodríGuez, A.: Algorithms for Maximum-likelihood Bandwidth Selection in Kernel Density Estimators, Pattern Recogn. Lett., Vol. 33, No. 13, pp. 1717–1724 (2012).

[32] Terrell, G. R. and Scott, D. W.: Variable Kernel Density Estimation, The Annals of Statistics, Vol. 20, No. 3, pp. 1236–1265 (1992).

[33] ÄĘwik, J. and Koronacki, J.: A combined adaptive-mixtures/plug-in estimator of multivariate probability densities, Computational Statistics & Data Analysis, Vol. 26, No. 2, pp. 199 – 218 (1997).

[34] Vincent, P. and Bengio, Y.: Manifold parzen windows, Advances in Neural Information Processing Systems, Vol. 15, pp. 825–832 (2002).

[35] López-Rubio, E. and de Lazcano-Lobato, J. O.: Soft clustering for non-parametric probability density function estimation, Pattern Recognition Letter, Vol. 29, No. 16, pp. 2085–2091 (2008).

[36] Girolami, M. and He, C.: Probability density estimation from optimally condensed data samples, IEEE Trans. Pattern Anal. Mach. Intell, Vol. 25, No. 10, pp. 1253–1264 (2003).

[37] Deng, Z., Chung, F. and Wang, S.: FRSDE: fast reduced set density estimator using minimal enclosing ball approximation, Pattern Recognition, Vol. 41, No. 4, pp. 1363–1372 (2008).

[38] Bottou, L.: Large-Scale Machine Learning with Stochastic Gradient Descent, Proceedings of the 19th International Conference on Computational Statistics, Paris, France, Springer, pp. 177–187 (2010).

[39] Bottou, L.: Stochastic Gradient Tricks, Neural Networks, Tricks of the Trade, Reloaded (Montavon, G., Orr, G. B. and Müller, K.-R., eds.), Lecture Notes in Computer Science (LNCS 7700), Springer, pp. 430–445 (2012).

[40] Fritzke, B.: A Growing Neural Gas Network Learns Topologies, Advances in Neural Information Processing Systems 7, MIT Press, pp. 625–632 (1995).

[41] Furao, S., Ogura, T. and Hasegawa, O.: An enhanced self-organizing incremental neural network for online unsupervised learning., Neural Networks, Vol. 20, No. 8, pp. 893–903 (2007).

[42] Sudo, A., Sato, A. and Hasegawa, O.: Associative Memory for Online Learning in Noisy Environments Using Self-Organizing Incremental Neural Network, IEEE Trans. Neural Networks, Vol. 20, No. 6, pp. 964–972 (2009).

[43] Makibuchi, N., Shen, F. and Hasegawa, O.: Online Knowledge Acquisition and General Problem Solving in a Real World by Humanoid Robots,

Proceedings of the 20th International Conference on Artificial Neural Networks, Thessaloniki, Greece, pp. 551–556 (2010).

[44] Okada, S. and Nishida, T.: Online incremental clustering with distance metric learning for high dimensional data, Proceedings of the International Joint Conference on Neural Networks, San Jose, California, pp. 2047–2054 (2011).

[45] Kankuekul, P., Kawewong, A., Tangruamsub, S. and Hasegawa, O.: Online incremental attribute-based zero-shot learning, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, pp. 3657–3664 (2012).

[46] Kawewong, A., Pimup, R. and Hasegawa, O.: Incremental Learning Framework for Indoor Scene Recognition, Proceedings of 27th AAAI Conference on Artificial Intelligence, Bellevue, Washington, pp. 496–502 (2013).

[47] Martinetz, T. and Schulten, K.: Topology Representing Networks, Vol. 7, No. 3, pp. 507–522 (1994).

[48] Hebb, D. O.: The Organization of Behavior: A Neuropsychological Theory, Wiley & Sons, New York (1949).

[49] Grossberg, S.: Classical and instrumental learning by neural networks, Progress in theoretical biology, Vol. 3, No. 51-141, pp. 42–47 (1974).

[50] Marr, D. and Thach, W. T.: A theory of cerebellar cortex, From the Retina to the Neocortex, Springer, pp. 11–50 (1991).

[51] Palm, G.: Neural Assemblies, an Alternative Approach to Artificial Intelligence, Springer-Verlag New York, Inc., Secaucus, NJ, USA (1982).

[52] Cooper, L. N.: A Possible Organization of Animal Memory and Learning, Proceedings of the Nobel Symposium on Collective Properties of Physical Systems (Lundquist, B. and Lundquist, S., eds.), Academic Press, New York, pp. 252–264 (1973).

[53] MacQueen, J.: Some methods for classification and analysis of multivariate observations, Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics, Berkeley, Calif., University of California Press, pp. 281–297 (1967).

[54] Vapnik, V. N.: Statistical learning theory, Wiley, New York (1998).

[55] Härdle, W.: Nonparametric and Semiparametric Models, Springer Series in Statistics, Springer-Verlag GmbH, Heidelberg (2004).

[56] Oyama, T., Githinji, K. S., Tsuge, S., Mitsukura, Y. and Fukumi, M.: Fast Incremental Algorithm of Simple Principal Component Analysis, IEEJ Transactions on Electronics, Information and Systems, Vol. 129, No. 1, pp. 112–117 (2009).

[57] Chin, T.-J. and Suter, D.: Incremental Kernel Principal Component Analysis., IEEE Trans. Image Processing, Vol. 16, No. 6, pp. 1662–1674 (2007).

[58] Takeuchi, Y., Ozawa, S. and Abe, S.: An Efficient Incremental Kernel Principal Component Analysis for Online Feature Selection, Proceedings

of the International Joint Conference on Neural Networks, Orlando, Florida, pp. 2346–2351 (2007).

[59] Nadaraya, E. A.: On estimating regression, Theory of Probability & Its Applications, Vol. 9, No. 1, pp. 141–142 (1964).

[60] Watson, G. S.: Smooth Regression Analysis, SankhyÄĄ: The Indian Journal of Statistics, Series A (1961-2002), Vol. 26, No. 4, pp. 359–372 (1964).

[61] Sung, H. G.: Gaussian mixture regression and classification, PhD Thesis, Rice University (2004).

[62] Friedman, N. and Koller, D.: Being Bayesian about network structure. A Bayesian approach to structure discovery in Bayesian networks, Machine learning, Vol. 50, No. 1-2, pp. 95–125 (2003).

[63] Wang, Z., Kuruoğlu, E. E., Yang, X., Xu, Y. and Huang, T. S.: Time varying dynamic Bayesian network for nonstationary events modeling and online inference, Signal Processing, IEEE Transactions on, Vol. 59, No. 4, pp. 1553–1568 (2011).

[64] Lepage, K. Q., Ching, S. and Kramer, M. A.: Inferring evoked brain connectivity through adaptive perturbation, Journal of computational neuroscience, Vol. 34, No. 2, pp. 303–318 (2013).

謝辞

本研究において，終始変わらぬ御指導を頂きました長谷川修准教授に心から感謝すると共に厚く御礼申し上げます．東京工業大学大学院総合理工学研究科知能システム科学専攻の渡邊澄夫教授，樺島祥介教授，新田克己教授，小野功准教授には，本研究に関して多くの有益なコメントを頂きました．ここに御礼申し上げます．また、日頃から研究について，建設的な議論をしてくれた長谷川研究室の皆様に感謝致します．なお，本研究の実施に当たり科学技術振興機構の戦略的創造研究推進事業からの支援を頂きました．記して感謝致します．最後に，研究活動を温かく見守り，ときに励ましてもくれた両親や兄弟，友人達に感謝いたします．