論文 / 著書情報
Article / Book Information

題目(和文)	マルチモーダル情報を用いたロボットコントロールシステムのための シンプルでロバストなアーキテクチャの提案と実ロボットにおける動作検証
Title(English)	
著者(和文)	秋川元宏
Author(English)	Motohiro Akikawa
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11735号, 授与年月日:2022年3月26日, 学位の種別:課程博士, 審査員:山村 雅幸,小野 功,青西 亨,瀧ノ上 正浩,長谷川 晶一
Citation(English)	Degree:Doctor (Engineering), Conferring organization: Tokyo Institute of Technology, Report number:甲第11735号, Conferred date:2022/3/26, Degree Type:Course doctor, Examiner:,,,,
学位種別(和文)	博士論文
Type(English)	Doctoral Thesis

学位論文

マルチモーダル情報を用いた
ロボットコントロールシステムのための
シンプルでロバストなアーキテクチャの
提案と実ロボットにおける動作検証

令和３年度博士（工学）申請

指導教員　山村雅幸　教授

氏名　　秋川元宏

論文要旨

人間と相互作用するロボットが開発されている。このようなロボットの制御システムは長い年月をかけて研究されている。古典的な制御方法では、システムの設計者がプログラムとして作り込むことでロボットの制御をしていた。近年では深層学習を用いた制御システムの開発が主流である。深層学習を用いた制御システムの中で状況認識の高精度化を目論み、マルチモーダル情報を入力とするものもみられる。しかし、マルチモーダル情報を入力とする深層学習を用いた制御システムには 2 つの欠点がある。第一に計算コスト、第二にノイズ耐性である。マルチモーダル情報を扱う深層学習はネットワークの規模が大きくなることで、計算コストが非常に高くなる。ネットワークの一部の結線を除去することで計算コストの増加を抑える研究も存在するが、どの結線を除去するかは自明ではなく、システム設計者に対し大きな負担を強いることになっている。ロボットは実環境において使役されるが、光源の位置が少し移動しただけで不安定になるになるなどの実環境由来の問題点が指摘されている。光源の位置のずれなどは環境ノイズと考えることができる。本研究では上記の 2 点を克服するロボット制御用アーキテクチャを提案する。アーキテクチャの提案に際し、アーキテクチャの特性を明らかにするために、擬似乱数によって生成された疑似データを用い学習および動作検証を行なった。さらに、リーチングタスクを行うアームロボットから得た実データを用いて学習実験を行い、実データに対しても提案アーキテクチャが動作可能であることを示した。

提案アーキテクチャは 3 つのモジュールで構成されている。最も重要なモジュールはアソシエイタユニットで、単一のホップフィールドネットワークで構成されている。このアソシエイタでマルチモーダル情報を統合し、記憶／想起を行う。もう 1 つのモジュールはエンコーダユニットである。エンコーダユニットは情報ごとに独立したエンコーダを持ち、3 層フィードフォワードニューラルネットワークで構成されている。本研究では画像、音声、アクチュエータの位置の３種類のマルチモーダル情報を統合するため、３つの独立したエンコーダがエンコーダユニットに存在することとなる。これらのエンコーダはセンサーから得たアナログ情報をホップフィールドネットワークが処理できるバイナリー情報へ変換する符号器として機能することとなる。最後のユニットがデコーダユニットである。デコーダユニットもエンコーダユニット同様に情報の種類ごとに独立した 3 層のフィードフォワードニューラルネットワークで構成されている。デコーダユニットはアソシエイタで処理されたバイナリー情報をシステムの出力となるようにアナログ情報への変換を行う復号器として機能する。学習

において、エンコーダとデコーダは情報の種類ごとのオートエンコーダとして同時に学習される。

擬似データを用いたアーキテクチャの動作検証実験において、提案アーキテクチャは単体でマルチモーダル情報を処理するオートエンコーダに比べ遥かに高いノイズ耐性を示した。また、提案アーキテクチャの高いノイズ耐性能力はアーキテクチャのサイズにより変化し、アーキテクチャのサイズが大きくなるにつれノイズ耐性が高くなることが判明した。

リーチングタスクを行うアームロボットから得た実データを用いた学習実験を行うために、動作検証実験で用いたアーキテクチャに変更を加えた。最も重要な変更点は、ホップフィールドネットワークのサイズ大きくし、時刻 t の状態および時刻 $t+1$ の状態を同時に銘記と想起できるようにしたことである。これにより、動作検証実験のアーキテクチャでは未実装であった、制御信号の生成というロボット制御システムにおける重要な機能を追加することができる。

実データを用いた学習実験では、3 方向、中間 9 状態のリーチングタスクを学習し、すべての方向へリーチングできることを示した。中間状態においても、軌跡およびシステムからの出力をアームロボットで再生したときの動作を目視で観察し、9 状態のうち、8 状態において、理想出力と遜色なく動作していることを確認した。

提案アーキテクチャは、オートエンコーダのように単一アーキテクチャでマルチモーダル情報を処理するアプローチと比べて、独立したエンコーダ、デコーダと計算コストの小さいホップフィールドネットワークで構成されていることから計算コストの削減に大きく貢献している。ノイズ除去能力の高いホップフィールドネットワークをアソシエイタとして採用することで、高いノイズ耐性を得ることができた。ただし、ホップフィールドネットワーク単体で扱った時に比べ、ノイズ除去能力は落ちている。ホップフィールドネットワークに銘記するパターンはエンコーダで自律的に生成されるため、必ずしも理想的なパターンとなっていないからだと考えられる。このように提案アーキテクチャはマルチモーダル情報を用いるロボット制御システムの基礎としては十分な能力を有するが、ロボットのリアルタイム制御を行うためには、エンコーダの改良が必要と考えられる。

目次

第 1 章：ロボット制御における技術

1.1 背景

1.1.1 人間と相互作用するロボットの開発

近年人間と相互作用するロボットの開発が盛んに行われている[1,2]。ASIOMO[3]、Pepper[4]、LOVOT[5]がその例である。ASIMO は二足歩行が可能で、人間が活動する空間を自由に歩行や旋回が可能であり、階段の上り下りも可能である。自律行動の観点で言えば自由度は高いが、人間との会話といった知的活動についてはあまり重きを置かれていないようである。Pepper は上半身が人型であり、下半身は車輪のロボットである。二足歩行型ではなく車輪であることから、人間が活動する空間を完全に共有すると言うことは難しい。Pepper の特徴は多彩なアプリケショーンを追加できることである。アプリケーションの追加により、身振り手振りを踏まえたコミュニケーションが可能である。この特徴から店舗などの受付などに利用されている。LOVOT はいわゆるペットロボットに分類される。移動は車輪であり、LOVOT 自体は人語を発話することはできない。現在開発されているロボットは自由に人間と相互作用できるわけではなく、制限付きの条件化でのみの相互作用に留まる。

1.1.2 作り込まれたプログラムによる制御

ロボットの制御を行う研究は長年続けられている[6,7]。その中で、プログラムを作り込むことでロボットの制御を行う研究は数多くある。有名なシステムとしてはセンス-プラン-アクト型[8]やサブサンプションアーキテクチャ[9]がある。センス-プラン-アクト型はその名の通り、センサーから情報得て、得た情報からどのような行動を行うかプランニングを行い、アクションを起こす。どのようにプランニングを行うかは設計者次第であり、如何にプランニングを作り込むかが肝となる。サブサンプションアーキテクチャは行動を単純なものへ分解し、分解された行動の組み合わせでロボットの制御を行う。分解の方法及び、分解された行動の組み合わせをどのようにプログラミングするかは設計者に依存する。

1.1.3 深層学習を用いたロボット制御

近年、深層学習などの機械学習を用いたシステム設計が主流となっている[10-14]。これは深層学習を用いた画像処理の分野における成功が発端となっている[15-18]。さらに分野として発展し[19-34]、深層学習の技術を用いることで、画像処理の分野だけではなく、音声認識や言語処理の分野において高精度な認識が可能となっている。深層学習における初期段階の研究ではユニモーダル情報を用いた研究がほとんどであった。しかし、より高い精度を得るために、画像と音声の組み合わせなどマルチモーダル情報を入力として用いたシステムも研究されている。マルチモーダル情報を扱うシステムは 2 種類に分類することができる。1 つ目は、入力はマルチモーダル情報であるが、同一時刻に処理するのは単一情報のみとするシステムである。このシステムは、情報の重要度など基にし、処理をする情報を切り替える[35,36]。もう 2 つ目は、マルチモーダル情報を統合し、同一時刻にまとめて処理するものである。現在主流となっているのは後者のシステムである。

深層学習における成果を受けて、ロボット分野においても深層学習の技術は応用されている。ユニモーダル情報を入力とするロボットへの応用は、物体認識などに深層学習を使うと言うものである[37]。しかし、物体認識が完了したあとの、行動選択には深層学習は使われず、ロボットごと特有のソフトウェアが使われている。深層学習を用いたロボット制御を含むシステムは、マルチモーダル情報を同時に処理するタイプのシステムがほとんどである[38-43]。これは、制御対象となっているロボットが複数のセンサーを搭載しているからである。比較的小さなロボットであるモバイルロボットでも、画像、超音波、ホールセンサーなど複数のセンサーが組み込まれている。Pepper に関して言えば、8 種類のセンサーを搭載している[44]。

マルチモーダル情報を入力として深層学習を用いる研究には、2 つの大きな課題がある。1 つ目は、ネットワークの最適化にかかるコストの増大である。このコストは、計算コストだけでなく、アーキテクチャを設計するためのコストでもある。マルチモーダル情報を単純に 1 つのニューラルネットワークの入力とすると、ネットワーク構造の肥大化により、計算コストが増大してしまう[14,15]。一部のシステムでは、制限されたネットワーク構造を採用する研究も存在するが、どのようにネットワークを制限すべきなのかは明らかになっていない。ネットワーク構造を制限することは、システム設計者の負担となっている。構造の異なるいくつかのモデルのネットワークの組み合わせで構成されるアーキテクチャを設計する場合、ネットワークがどのように他のネットワークと接続されるかが最も重要であるが、これも明らかではない。また、次元圧縮されたデータを

扱うのか、センサーから受け取った信号を直接扱うのかなど、アーキテクチャが入力として何を得るかも考慮する必要がある。Ramachandram と Taylor [45] は、設計者が構造に多くの注意を払わなければならないことから、否定的に「アーキテクチャの設計は科学というよりも芸術である」と述べている。しかし、アーキテクチャの設計は科学であるべきである。理由なくして、アーキテクチャの拡張やタスクごとへの適応は十分にできない。

２つ目はノイズ耐性である。ロボット制御では時刻ごとに状況が変化し、その変化に応じて出力を得る必要があるため、システムは時系列データを扱う。複数のフレーム情報を 1 つにまとめ、時系列データとして入力し、処理するシステムがある[38]。このシステムは、アクチュエータの動き始めや、ノイズとして扱われるべき照明の変化などの環境変化がすべての入力に含まれるまでは不安定である。この不安定さの原因は、ノイズ除去機構がアーキテクチャに実装されていないことが原因と考えられる。現在の深層学習を用いたシステムは、ニューラルネットワークの関数近似能力と汎化性能を用いて、特徴量に基づいた出力を得る方法を取っている。ノイズを除去する仕組みとし、損失関数に項を追加したり、学習データにノイズを加えたりして、ノイズ耐性を高めるといった手段を採用する場合もあるが、これらのアルゴリズムは想定されるノイズの範囲内でしかノイズを除去することがでない。そのため、想定外のノイズに対しては無防備であり、どのような結果が出力されるのかは定かではない。ノイズ除去の方法の提案とそれを回避する手段のいたちごっこは画像処理の分野で特に研究されているが[46]、解決の決定打になる研究はまだ発表されていないため、ロボット分野での応用も限定的である。

1.2 研究目的と論文の構成

本研究では，先述したコストとノイズ耐性の２つの問題を考慮したロボット制御用アーキテクチャである、Memorizing and Associating Converted Multimodal Signal Architecture（MACMSA）を提案することが主な目的とする。2 章において、MACMSA の基本構造の提案および最適化方法の説明を行う。3 章において、擬似データを用いた内部解析を行い、MACMSA の素性を明らかにする。4 章において、実ロボットから得たデータを扱うために MACMSA の改良を行う。この改良を踏まえ、リーチングタスクを行うロボットから得られたデータを用いて、改良された MACMSA が実データを学習できるか検証を行う。また、この検証の一環として、システムからの出力を実際に実ロボットアーム上で実行し、理想的な挙動と比較を行う。5 章において、総合討論として、MACMSA を利用することの優位性と、MACMSA の改善すべき点を議論する。

第２章：マルチモーダル情報を統合するアソシエイティブメモリを含むシステムの提案

2.1 導入

本章ではマルチモーダル情報を統合するアーキテクチャとして Memorizing and Associating Converted Multimodal Signal Architecture(MACMSA)を提案し、アーキテクチャの詳細と最適化手順を述べる。

2.2 手法提案

2.2.1 提案手法の概要

提案されるアーキテクチャの概要は図 2-1 に示す通りである。MACMSA はエンコーダ、アソシエイタ、デコーダの 3 モジュールで構成される。アソシエイタが提案アーキテクチャにおいて最も重要なモジュールであり、マルチモーダル情報を統合する役目を果たす。アソシエイタはホップフィールドネットワークにスパース化を施したものを採用する。ホップフィールドネットワークを採用することにより強力なノイズ耐性を実現する。また、スパース化を施すことにより記憶容量の向上が望める。エンコーダはロボットに搭載されているセンサーから得た実数値または整数値からなる情報をホップフィールドネットワークが処理できる 2 値へと変換する役割を持つ。デコーダはアソシエイタで処理された 2 値を実数値へと変換する役割を持つ。エンコーダとデコーダは情報の種類ごとに独立した5層のオートエンコーダとして最適化される。最適化完了後、エンコーダ、デコーダとして 3 層のフィードフォワードニューラルネットワークに分割される。本研究ではロボットに搭載されたセンサーから得る情報として、画像、フーリエ変換された音声、アクチュエータの位置情報を取得することを想定しており、この 3 種類の情報を統合するものとする。各種類の情報はエンコーダにより 2 値へ変換される。その後各エンコーダからの出力はすべて結合され、1 つのベクトルとなる。結合されたベクトルはアソシエイタであるホップフィールドネットワークへ初期状態として与えられる。このとき事前に銘記されたパターンに対し符号反転ノイズがベクトルに含まれていた場合はホップフィールドネットワークの機能により符号反転ノイズは修正される。想起されるパターンはホップフィールドの定常状態として出力される。想起されたパターンは情報の種類ごとに分解され、各デコーダへ入力される。最終的に各デコー

ダからノイズが除去された出力を得ることができる。単に各情報を結合し入力とするオートエンコーダとの大きな違いはアソシエイタによる強力なノイズ除去機構を有するかどうかである。

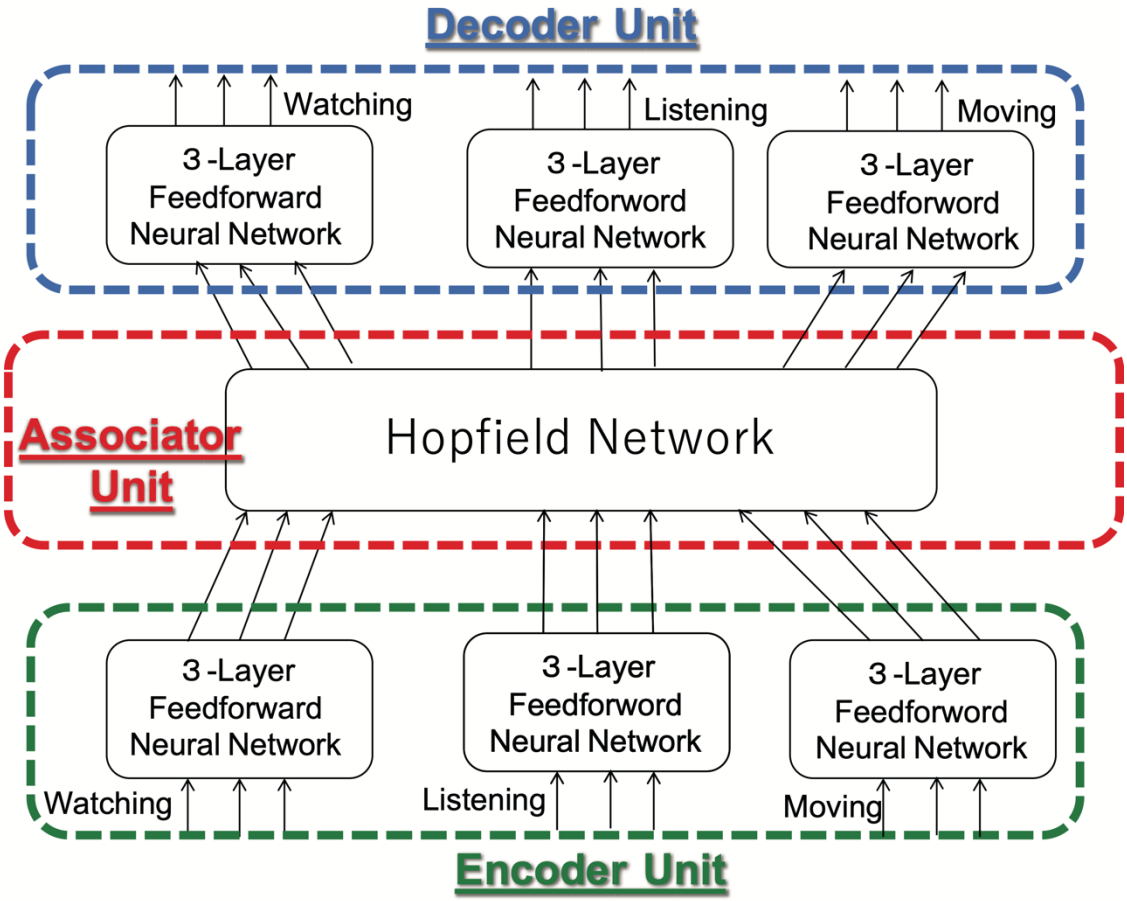

図 2-1: Memorizing and Associating Converted Multimodal Signal Architecture の概要図

2.2.2 オートエンコーダの最適化原理

オートエンコーダは深層学習における初期に提案されたネットワークである[47]。図 2-2 は 5 層で構成されるオートエンコーダである。オートエンコーダは入力と同じものを出力するように最適化を行うことにより、教師信号を別途準備する必要がないことが特徴である。入力自身を符号化することから日本語では自己符号化器と呼ばれる。オートエンコーダは次元圧縮が目的として使用されることが多い。そのため、構造としては中心の中間層で折り返しした、砂時計のような構造をとることが多い。各層では次元圧縮された特徴が学習されることとなるため、特徴検出器としても機能する。動作原理および最適化方法は一般的なフィードフォワードニューラルネットワークと同様である。1 ニューロンに着目した時、ニューロンからの出力は次の式で求められる[48]。

$$y = f\left(\sum_{\mathrm{i}=1}^{N} w_{\mathrm{i}} x_i\right) \tag{2-1}$$

$$f(u) = \frac{1}{1 + e^{-au}} \tag{2-2}$$

式 2-1 におけるNは着目したニューロンが受け取る入力の数である。wはニューロン間の重みを表し、xは入力の値を表している。$f()$は活性化関数と呼ばれる関数で、ニューラルネットワークの使用目的に応じて変更される。式 2-2 は回帰問題で活性化関数としてよく使われるシグモイド関数を表している。aはゲインと呼ばれ、シグモイド関数の立ち上がりを決めるパラメータである。そのほかの活性化関数としては、ReLU 関数や多クラス分類問題で使われるソフトマックス関数などがある[49]。

オートエンコーダでは目標となる教師信号である入力との誤差が小さくなるように重みwを調整する。誤差を表現する関数のことを損失関数と呼ぶ。損失関数としてよく使われるのは二乗和関数や平均二乗誤差関数である。本節では二乗和関数を例に説明を行う。二乗和関数は式 2-3 で表される。

$$E(w) = \sum_{n} \|\boldsymbol{y}_n - \boldsymbol{t}_n\|^2 \tag{2-3}$$

$\boldsymbol{y}$はニューロンからの出力を表しており、$\boldsymbol{t}$は教師信号を表している。この損失関数に対し重みwで微分を行うことで、wの修正量を決定する。多層のニューラルネットワークにおいては、出力層から入力層に向かって、誤差を伝播させていき、出力層から逆方向に向かって重みwの最適化を行う。この方法のことを誤差

逆伝播法と呼ぶ。誤差逆伝播法を用いると式 2-3 の微分式は式 2-4 のように表すことができる。

$$\frac{\partial E}{\partial u_j^{(l)}} = \sum_k \frac{\partial E}{\partial u_k^{(l+1)}} w_{kj}^{(l+1)} f'\left(u_j^{(l)}\right) \tag{2-4}$$

u_kとu_jはk番目およびj番目のニューロンにおける誤差を表している。lおよび$l+1$は層の番号を示している。$f'()$は活性化関数の逆関数を意味する。

二乗和関数や平均二乗誤差関数などの損失関数に、目的に応じて、項が追加されることがある。追加される項としてよく使われるのは、重みに制限を加える項である。そのほかにも、出力のスパース化図るためにカルバック=ライブラーダイバージェンスが項として付け加えられることもある。

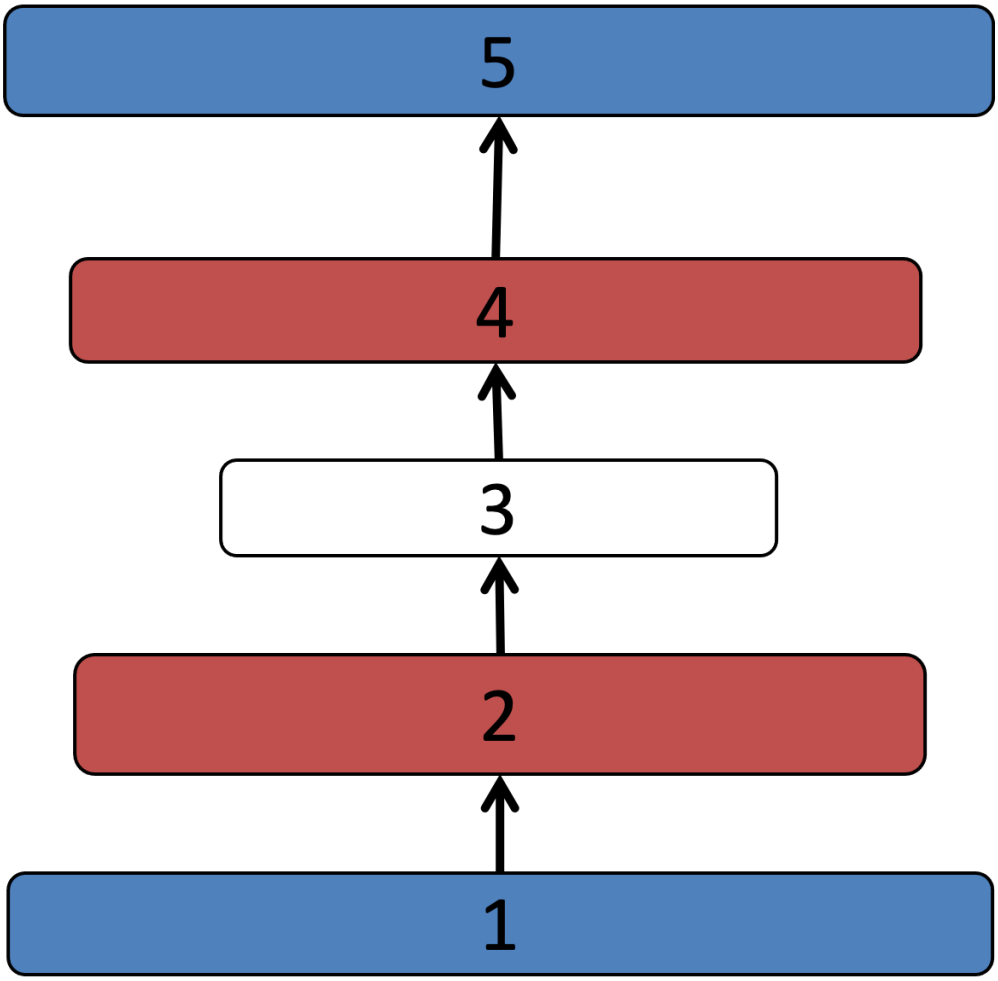

図 2-2:5 層のオートエンコーダ

2.2.3 ホップフィールドネットワークにおける銘記と想起

ホップフィールドネットワークは相互結合型のニューラルネットワークである[50]。ホップフィールドネットワークは連想記憶としての機能を持つ。ホップフィールド以外の連想記憶を行うネットワークも存在する[51-55]。ホップフィールドネットワークの用語、動作原理、最適化方法は、オートエンコーダを含むフィードフォワードニューラルネットワークと異なる。これは、フィードフォワードニューラルネットワークが神経回路の模倣から研究が開始されたのに対し、ホップフィールドネットワークは電子スピンにおける場のエネルギー計算を目的として研究が開始したからである[56]。ホップフィールドネットワークは電子スピンの方向を処理するモデルであるであるため、扱うデータは–1 と 1 の 2 値で構成されるパターンとなる。フィードフォワードニューラルネットワークにおける入力に対応するものは、ホップフィールドネットワークでは初期状態と呼ばれる。また、フィードフォワードニューラルネットワークにおける出力に対応するものは、ホップフィールドネットワークでは定常状態と呼ばれる。ネットワークを流れる情報の流れも異なる。フィードフォワードニューラルネットワークは一方向にのみ情報が流れるのに対し、ホップフィールドネットワークは、ニューロン間を行き来する。これにより、ホップフィールドネットワークは 1 行程で定常状態へと状態を推移させることができない。ホップフィールドネットワークは初期状態として内部状態を受け取り、内部状態の更新を繰り返し行い、内部状態の変更がなくなった状態を定常状態として扱う。最適化における用語も異なる。ホップィールドネットワークでは学習のことを明記と呼ぶ。ホップフィールドネットワークへの明記はフィードフォワードニューラルネットワークの学習とは異なり、明記されるデータが決まれば重みが一意に決まる。銘記には式 2-5 を用いて銘記される[57]。

$$J_{ij} = \frac{1}{M}\sum_{\mu=1}^{M} \zeta_i^{\mu}\zeta_j^{\mu} \tag{2-5}$$

J_{ij}はi番目とj番目間のニューロンの重みである。Mは銘記するパターンの数を表す。μはパターンのインデックスを表す。

初期状態を受け取り、定常状態へと状態を推移させることを想起と呼ぶ。想起は次の式 2-6 と式 2-7 で行うことができる。

$$x_i^{t+1} = F\left(\sum_{j\neq i}^{N} J_{ij}\, x_j^{t}\right) \tag{2-6}$$

$$F(u) = sgn(u) \tag{2-7}$$

x_j^tは時刻 tでの内部状態を表しており、Nはホップフィールドネットワークを構成するニューロン数である。式 2-7 における$sgn()$は内部状態の符号を出力する関数である。uは内部状態を表す。式 2-5 から式 2-7 はパターンを構成する−1 と 1 の割合が同じときに適応される式である。

−1 と 1 の割合が同じではないパターンを扱うホップィールドネットワークをスパースホップフィールドネットワークと呼ぶ。ホップィールドネットワークをスパース化することで記憶容量が向上することが知られている。スパースホップフィールドネットワークにおける銘記を行う式は式 2-8[57]となる。

$$J_{ij} = \frac{1}{M}\sum_{\mu=1}^{M}(\zeta_i^\mu - a)(\zeta_j^\mu - a) \tag{2-8}$$

aはパターンの−1 と 1 の割合を示す平均発火率であり、実際にパターンから計算される値である。想起を行う式は次に示す 2 つの式に変化する。

$$x_i^{t+1} = F\left(\sum_{j\neq i}^{N} J_{ij}\, x_j^t + h\right) \tag{2-9}$$

$$F(u) = sgn(u) - b \tag{2-10}$$

hはシフトと呼ばれ、$h = a(1-a^2)$の時に信号の最大化が行われる。bはバイアスと呼ばれ、$b = a$の時クロストークノイズを最小化することができる。

スパースホップィールドネットワークの記憶容量は式2-11 を用いて近似的に求めることができる[57]。

$$\alpha_C(a) = \frac{\alpha_C(0)}{(1-a^2)} \tag{2-11}$$

ここで、$\alpha_C(0) = 0.138N$であり、Nはホップィールドネットワークを構成するニューロン数である。

2.2.2 提案手法の学習手順

提案手法は次の 3 ステップで最適化が行われる。(1)エンコーダとデコーダの最適化、(2)アソシエイタへ銘記する 2 値パターンの生成、(3)アソシエイタへのパターンの銘記。本節では各手順を順に説明する。

2.2.2.1 エンコーダとデコーダの最適化

エンコーダとデコーダは3層フィードフォワードニューラルネットワークとして構成されているが、最適化においてはエンコーダの出力層とデコーダの入力層を共通とし、5 層のスパース化されたオードエンコーダとして学習を行う(図 2-3)。

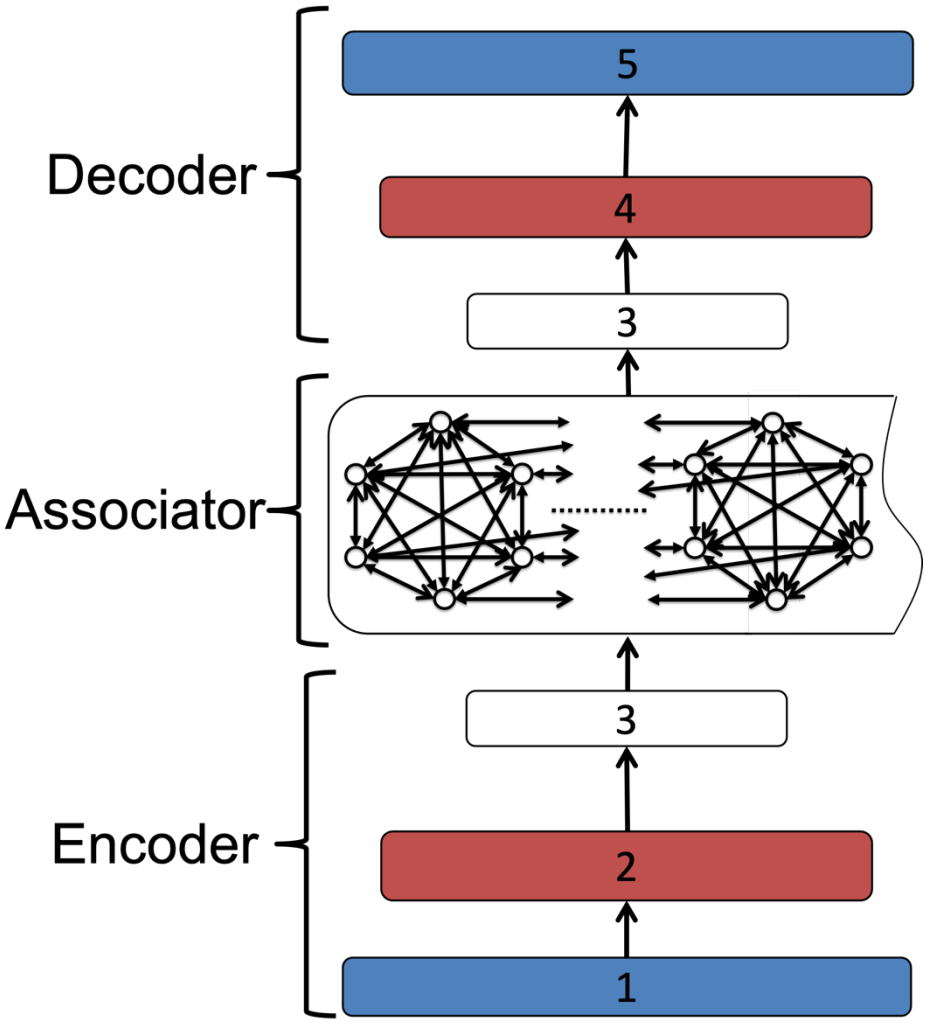

図 2-3:オートエンコーダとして学習されるエンコーダおよびデコーダ

オートエンコーダを用いたシステムの研究は多く存在し[58-65]、そのほとんどは特徴量の検出または次元圧縮を目的としてオートエンコーダが用いられる。本研究ではエンコーダの出力層となるオートエンコーダの 3 層目を除き、ゲイン値を 1.0 とした標準シグモイド関数を活性化関数として用いる。エンコーダの出力層となるオートエンコーダの 3 層目は最適化の進行状態に応じゲイン値が徐々に高くなるようにし、最適化完了時には限りなく 2 値を出力するようにする。これはアソシエイタであるホップフィールドネットワークが 2 値を扱うモデルであるため、アソシエイタへの入力となるエンコーダの出力が限りなく 2 値である必要があるからである。最適化手法は Stochastic Gradient Descent (SGD)を用いた逆誤差伝播法[48]を採用する。先述したように本研究では、オートエンコーダにおける 3 層目のシグモイド関数のゲイン値を徐々に高くなるように最適化を行う。この最適化方法は図 2-4 で示すように、逆誤差伝播法とゲインの再設定を繰り返し行うことにより実現する。

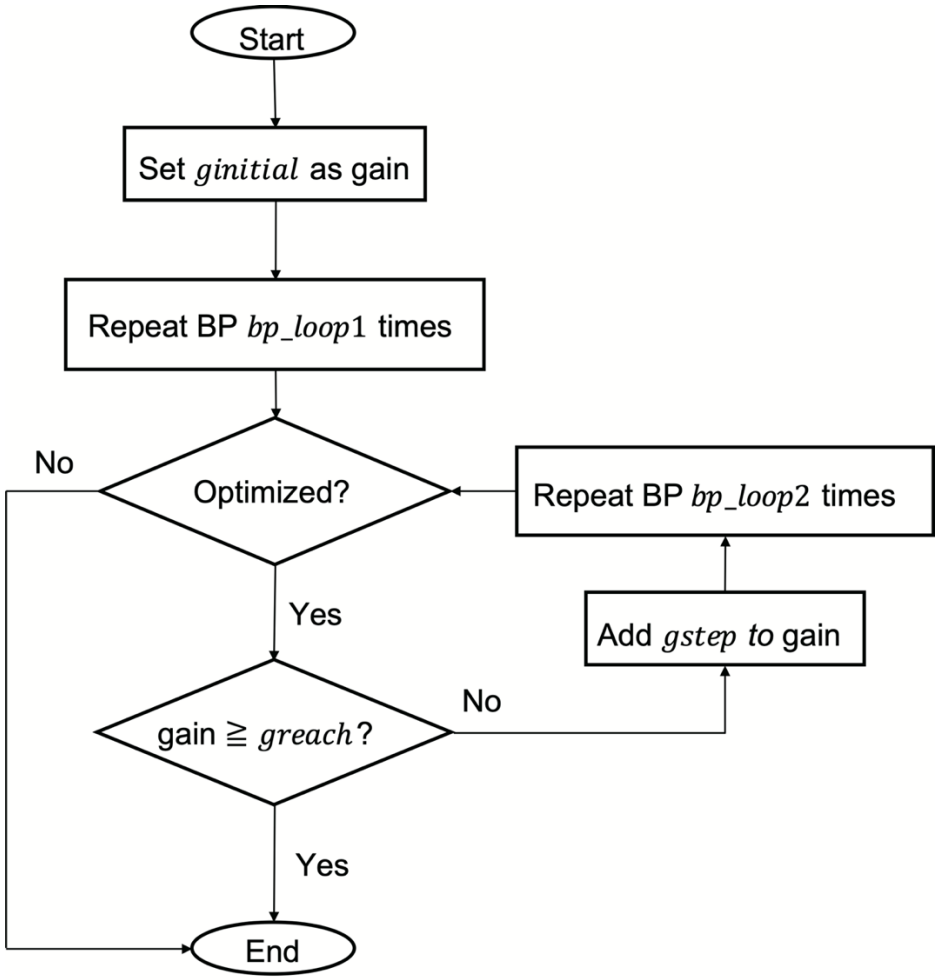

図 2-4:オートエンコーダの最適化中のゲイン値設定手順

本研究で用いる損失関数は式 2-12 で表される。

$$E(w) = \sum_{n} \|\boldsymbol{y}_n - \boldsymbol{t}_n\|^2 + \beta \sum_{j} KL(\rho||\hat{\rho}_j) \tag{2-12}$$

右辺第 1 項は学習データである$\boldsymbol{t}$とオートエンコーダの出力である$\boldsymbol{y}$の誤差を示す項となり、二乗和となっている。nは学習データのインデックス番号である。右辺第 2 項はスパース強度に関する項でありカルバック=ライブラーダイバージェンスである[66]。βはスパース項の影響度を調整するパラメータであり、ハイパーパラメータである。ρはニューロンの発火率の目標値であり、この値もハイパーパラメータである。$\hat{\rho}$は実際に計測された発火率である。jはニューロンのインデックス番号である。ρを調整することで、出力の発火頻度を調整することができるので、アソシエイタに銘記するパターンの基となる 0 と 1 の割合を調整することができる。

オートエンコーダの重み修正に用いられる微分式は式 2-12 を基にし、式 2-13 で表される[67]。

$$\frac{\partial E}{\partial u_j^{(l)}} = \left\{ \sum_{k} \frac{\partial E}{\partial u_k^{(l+1)}} w_{kj}^{(l+1)} + \beta \left(-\frac{\rho}{\hat{\rho}_j} + \frac{1-\rho}{1-\hat{\rho}_j} \right) \right\} f'\left(u_j^{(l)}\right) \tag{2-13}$$

2.2.2.2 アソシエイタへ銘記する 2 値パターンの生成とホップフィールドへの銘記

全種類の情報かつ全学習データを前節で最適化されたデコーダを用いて、近似された 0 と 1 で構成されるベクトルへ符号化を行う(図 2-5)。ホップフィールドネットワークに銘記するパターンは完全な 2 値かつ{−1,1}であるため、近似された 0 と 1 からなるベクトルを閾値 0.5 とし、閾値以下の値を−1、閾値以上の値を 1 に置き換えることで、{−1,1}で構成されたベクトルへ置き換える。この近似された 0 と 1 から{−1,1}への置き換えはシステムの最適化時のみだけでなく、推論時も行われることに注意されたい。

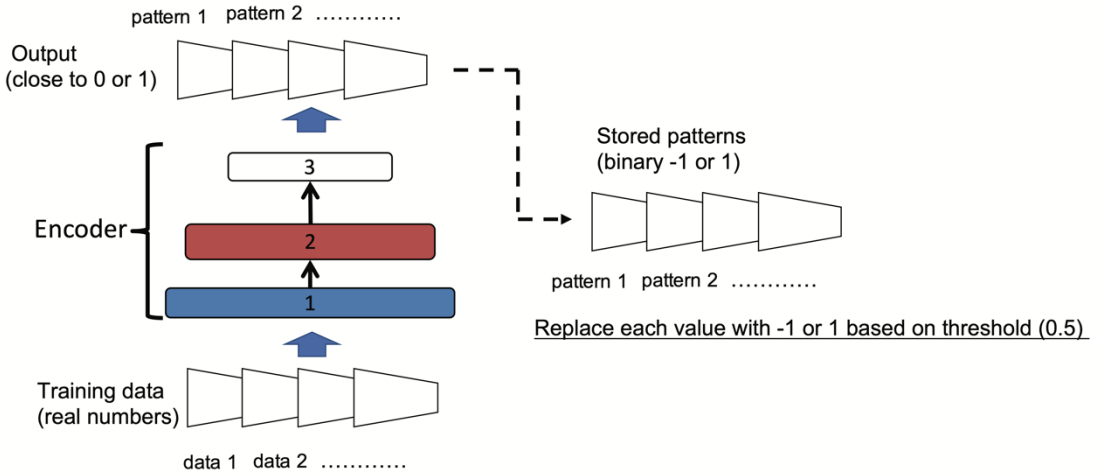

図 2-5:近似された 0 と 1 から−1、1 への置き換え

{−1,1}へ置き換えられた全情報は学習データのインデックス番号ごとに結合された状態で式 2-8 を用いてホップフィールドネットワークへ銘記される。

2.2.3　推論時のデータフロー

3 種類のセンサーから得た情報は各エンコーダを用いて近似された 0 と 1 で構成されたベクトルへと符号化される。符号化された 3 つのベクトルはそれぞれの要素に対し、0.5 を閾値として、−1 と 1 へ置き換えられる。{−1,1}で構成された 3 つのベクトルは 1 つのベクトルへと結合される。この時、入力にノイズが含まれていた場合、ノイズは銘記されたパターンに対し符号反転ノイズとしてエンコーダからの出力に現れる。結合されたベクトルはホップフィールドネットワークへ初期状態として与えられる。ホップフィールドネットワークは内部状態の更新を行い、定常状態を得る。この時、符号反転ノイズはホップフィードネットワークの特性により修正され、明記されたパターンが想起される。定常状態として得られたベクトルを情報の次元と合うように、3 つのベクトルに分割する。すべてのベクトルに対し、0.0 を閾値に−1 を 0 に、1 をそのまま 1 に置き換えを行う。各デコーダを用いて、{0,1}で構成されたベクトルを実数値へ復号する。この復号化された実数がシステムからの出力となる。このデータフローを図示したものが図 2-6 となる。

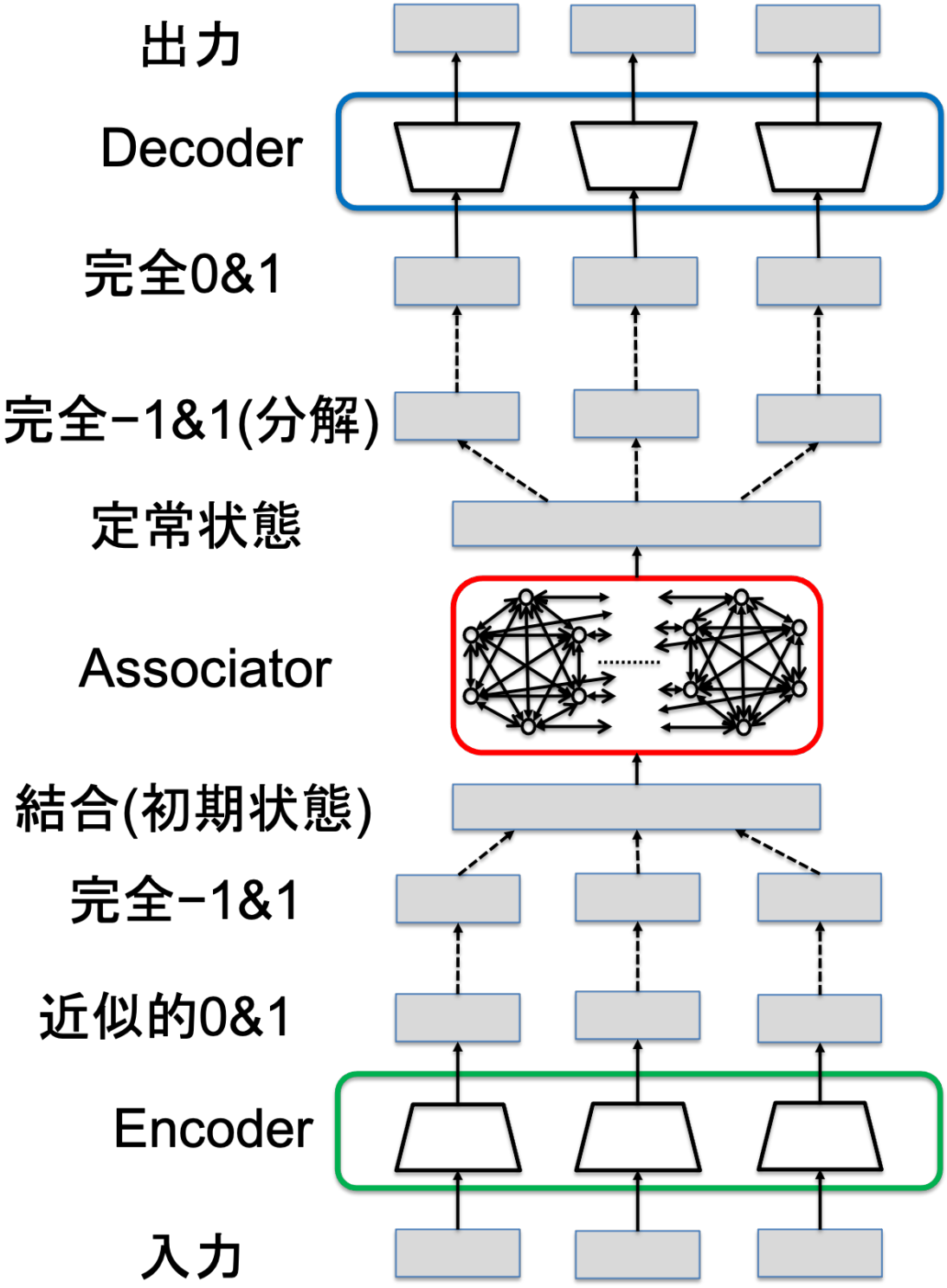

図 2-6:推論時のデータフロー

第 3 章:提案手法における性能と挙動検証

3.1 導入

本章では MACMSA における性能と挙動の検証を行う。検証は大きく分け、アーキテクチャを構成するモジュール単位での動作検証、単一でマルチモーダル情報を処理するオートエンコーダとの性能比較、アーキテクチャを構成するネットワークのサイズを変更した時の挙動検証、ノイズが不均等にシステムへ入力された時の挙動検証の 4 つである。これらの検証により、MACMSA の詳細な素性を理解することができる。

3.1.1 モジュール単位での動作検証

本節では、最適化を行なった提案手法のモジュールごとの性能と動作検証を、擬似データを用いて行う。評価項目は(1)最適化時の評価値の変化、(2)オートエンコーダとしての性能、(3)アソシエイタでの想起率、(4)システムにおけるノイズ除去性能の 4 項目(図 3-1)である。

第 1 項目ではエンコーダとデコーダとなるオートエンコーダの最適化において、評価値が誤差逆伝播法の繰り返し回数に対しどのように変化していくかを観察する。第 2 項目ではエンコーダとデコーダとなるオートエンコーダ単体での性能を観察する。第 3 項目ではエンコーダとアソシエイタを組み合わせ、アソシエイタにおいてのノイズ除去率を想起率として観察する。第 4 項目ではシステム全体としてのノイズ除去性能を観察する。

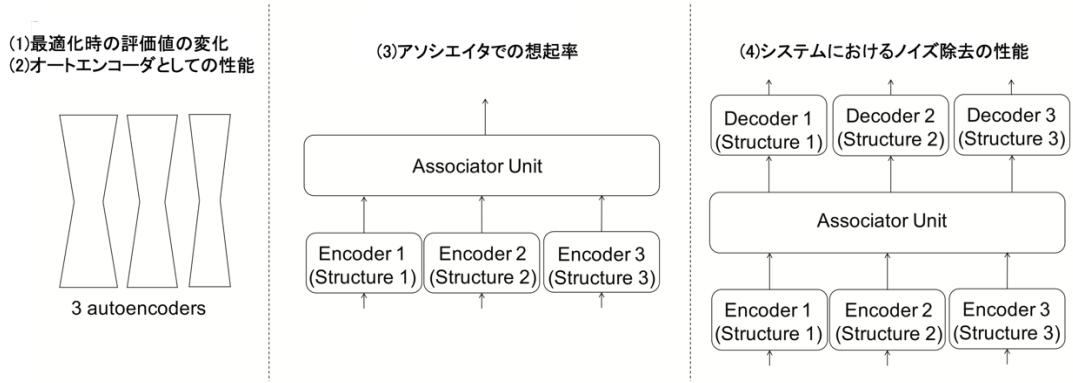

図 3-1:各実験において対象となるモジュール

(1)最適化時の評価値の変化

3 つのオートエンコーダの最適化を行い、損失関数がどのような変化を経て、ネットワークが最適化されていくかを観察する。本研究ではセンサーから得ら

れる情報の次元数が異なることを想定し、表 3-1 のように 3 種類のネットワークサイズを最適化することとする。入力層において、最も大きい構造と最も小さい構造ではニューロン数が 10 倍違うことに注意されたい。中間層については、それぞれ第 1 隠れ層を入力ニューロン数の 0.75 倍、第 2 隠れ層を 0.5 倍とする。学習データは正規分布に従う擬似乱数として生成される擬似データ 10 件とする。アーキテクチャが様々な平均、標準偏差を持つデータを学習できることを示すために、平均 0.25、0.5、0.75 の 3 種類、標準偏差 0.25、0.28、0.31 の 3 種類の擬似データを用いる。ただし、生成範囲は[0.0,1.0]とする。平均、標準偏差の組み合わせとして 9 種類あり、3 つのオートエンコーダに対してすべての組み合わせを適応するため、合計で 729 通りの組み合わせを実験することになる。さらに、オートエンコーダの初期重みを決定する乱数シードを変えて 10 回実験を行う。このため、学習は合計で 7290 回行われることになる。オートエンコーダの初期重みは平均 0.0、標準偏差 0.3 の正規乱数を用いて初期化される。エンコーダは情報の次元圧縮が目的の 1 つであるため、学習するデータの数によっては、中間層のニューロン数をより少なくすることができる。しかし、各エンコーダの第 2 隠れ層のニューロン数の合計がホップフィールドネットワークのニューロン数となり、ホップフィールドネットワークのニューロン数は記憶容量に直接影響するため、慎重に決定する必要がある。本実験では、学習データ数に対し、十分な記憶容量になるようにニューロン数を設定してある。

誤差逆伝播法の繰り返し回数であるbp_loop1およびbp_loop2は表 3-1 に示す通りである、また、そのほかの最適化に関するパラーメタは表 3-2 の通りである。

bp_loop1およびbp_loop2は各ゲイン値において評価値が十分に収束する値にする必要がある。bp_loop1およびbp_loop2が不十分な値の場合は、ゲイン値の変更時に勾配消失が起こり、最適化が失敗してしまう可能性がある。$gstep$が 1.0 を超えるような大きな値になると、ゲイン値の変更時にネットワーク構造の変化が大きく、最適化が失敗する可能性がある。逆に小さすぎると、計算コストの増加を招く。学習率αが大きいと最適化は速くなるが、十分に収束しない可能性がある。逆に小さくしすぎると計算コストの増加を招く。ただし、本研究では収束することを重要視し、十分に小さな値を設定した。ρを大きくするとホップフィールドネットワークに明記するパターンがよりスパース化されるため、記憶容量の向上に繋がる。高くしすぎるとホップフィールドネットワークでのノイズ耐性が低下を引き起こす可能性がある。そのため、記憶させる学習データ数とホップフィールドネットワークの記憶容量を考慮し、できる限り低い値を設定した方が良いと考えられる。βを高くすると、学習データそのものを特徴量として学習を行なってしまう[67]。一方、低い値にすると雑然とした特徴量を学習し

てしまう。本実験で使われるβ近辺に設定することで、ホップフィールドネットワークに明記することができるパターンを生成することができるようになる。

表 3-1:ネットワーク構造と最適化繰り返し回数

Autoencoder structure	Input layer	Hidden layer 1	Hidden layer 2	Hidden layer 3	Output layer	*bp_loop1*	*bp_loop2*
Structure 1	1 000	750	500	750	1000	15 000	7 500
Structure 2	500	375	250	375	250	10 000	5 000
Structure 3	100	75	50	75	100	7 500	3 750

表 3-2:最適化に係るパラメーター

ginitial	*greach*	*gstep*	α	ρ	β
1.0	10.0	0.5	0.001	0.4	0.2

(2)オートエンコーダとしての性能

最適化されたオートエンコーダに対し、ノイズを含む入力を加え、オートエンコーダ自体のノイズ耐性を観察する。オートエンコーダに入力されるノイズを含むデータは、学習データを基準とし、学習データの一部が乱数で書き換えられたデータとする。ノイズの混入率は 0%から 50%の間で 2%区切りとする。x%のノイズ混入率の場合、次元数のx%に相当する要素がランダムで選択され、選択された要素の値は[−1,1]の範囲で一様乱数を用いて生成される。ノイズ混入率は 3 つすべての情報に対し同一の混入率とする。具体例として、25%のノイズ混入率の場合、1 つ目の情報に対しては、次元数である 1000 のうち 250 要素が書き換えられ、2 つ目の情報に対しては次元数 500 のうち 125 要素が書き換えられ、3 つ目の状態に対しは、次元数 100 のうち 25 要素が書き換えられることとなる。ノイズ混入率 0%は学習データと全く同じものが入力されることを意味する。同一アーキテクチャに対する実験はノイズ混入率ごとに書き換えられる要素を変更し、10 回試行する。

(3)アソシエイタでの想起率

前節で最適化されたエンコーダとアソシエイタを組み合わせ、アソシエイタにおける想起率を観察する。「オートエンコーダとしての性能」の実験と同様にエンコーダへの入力は学習データと一致するノイズ 0%から最大ノイズ混入率 50%で観測が行われる。また、エンコーダへの入力に対し、書き換えを行う要素を変え、10 回試行することも「オートエンコーダとしての性能」の実験と同様である。

(4)システムにおけるノイズ除去の性能

最適化されたシステム全体を通し、ノイズ耐性およびノイズの除去性能を観察する。これまでの実験と同様にエンコーダへの入力は学習データと一致するノイズ 0%から最大ノイズ混入率 50%で観測が行われる。また、システムへの入力に対し、書き換えを行う要素を変え、10 回試行することもこれまでの実験と同様である。

3.1.2 提案手法とマルチモーダル情報を処理する単一オートエンコーダの比較

提案手法と、マルチモーダル情報を処理するオートエンコーダでノイズ耐性に対する違いを観察する。提案手法のネットワーク構成および、最適化に関するパラーメタは前節と同じである。単一オートエンコーダでは 3 種類の情報を結合したものを入力として得る(図 3-2)。オートエンコーダのネットワークサイズは、

前節における各ネットワークのニューロン数(表 3-1)の和と同じに設定する。つまり、オートエンコーダの入力層は 1600 個のニューロン、隠れ層 1 は 1200 個のニューロン、隠れ層 2 は 800 個のニューロンを持つことになる。誤差逆伝播法の繰り返し回数は評価値を完全に収束させるために、75,000 回とした。また、このオートエンコーダの活性化関数におけるゲイン値はすべて 1.0 とした。

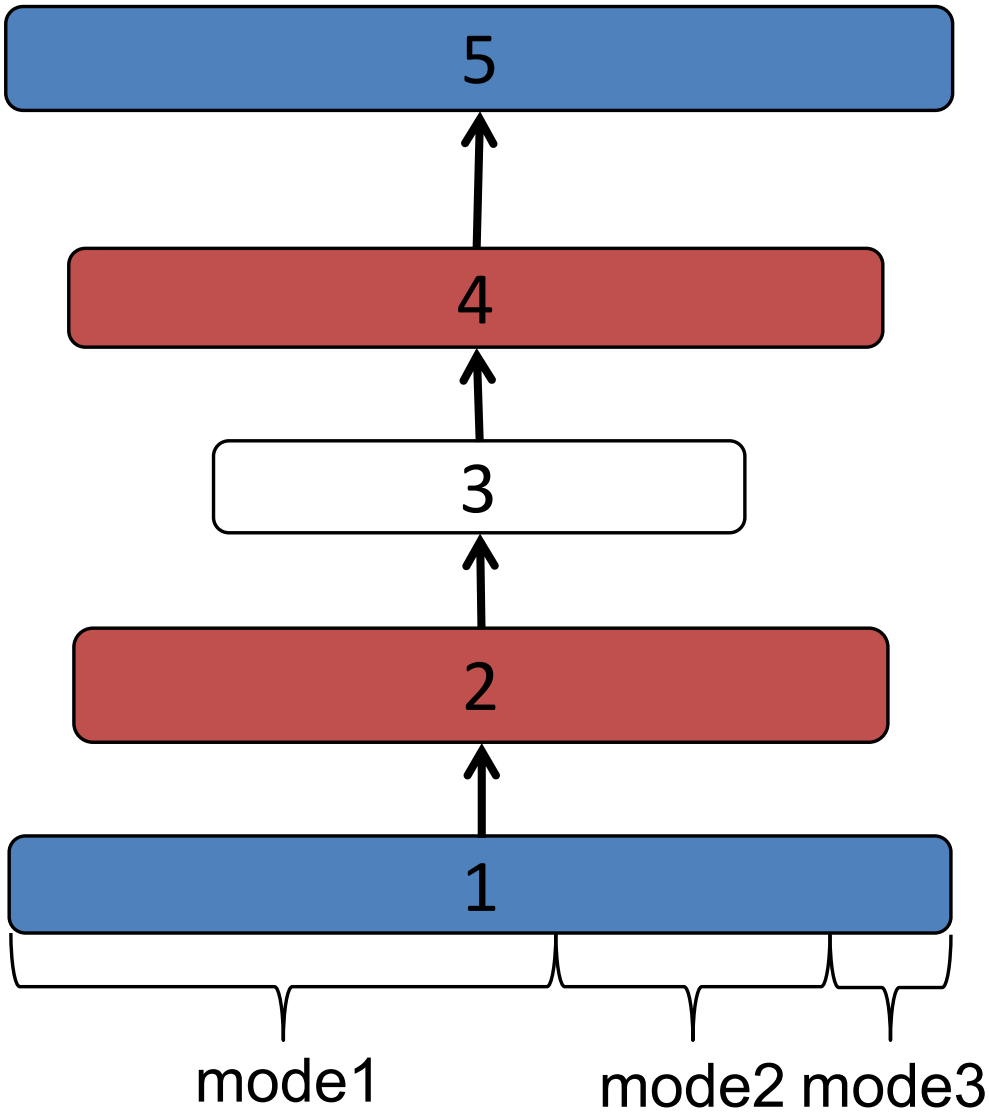

図 3-2: 3 つの情報を処理する単一オートエンコーダ

3.1.3 提案手法のネットワークサイズとノイズ耐性

提案手法のネットワークサイズを変化させたときの、ノイズ耐性の変化を観察する。ネットワークサイズの異なる 4 種類のアーキテクチャ(表 3-3)に対し、

3.1.1 節で行った「オートエンコーダとしての性能」、「アソシエイタでの想起率」、「システムにおけるノイズ除去の性能」と同様の実験を行う。

表 3-3:4 種類のアーキテクチャと各最適化繰り返し回数

Architecture	Autoencoder Structure	Input layer	Hidden layer 1	Hidden layer 2	Hidden layer 3	Output layer	***bp_loop1***	***bp_loop2***
Arch 1	Structure 1	100	75	50	75	100	7500	3750
	Structure 2	70	53	35	53	70	7500	3750
	Structure 3	40	30	20	30	40	7500	3750
Arch 2	Structure 1	180	135	90	135	180	7500	3750
	Structure 2	120	90	60	90	120	7500	3750
	Structure 3	60	45	30	45	60	7500	3750
Arch 3	Structure 1	360	270	180	270	360	7500	3750
	Structure 2	240	180	120	180	240	7500	3750
	Structure 3	120	90	60	90	120	7500	3750
Arch 4	Structure 1	1000	750	500	750	1000	15 000	7500
	Structure 2	500	375	250	375	250	10 000	5000
	Structure 3	100	75	50	75	100	7500	3750

最も大きなネットワークは 3.1.1 節で実験対象としたアーキテクチャサイズと同じである。その他のアーキテクチャサイズはアソシエイタであるホップフィールドネットワークの記憶容量を考慮し決定した。表 3-3 における最も小さいアーキテクチャサイズより小さいアーキテクチャは提案手法では最適化ができなかった。各ネットワークサイズの誤差逆伝播法の繰り返し回数は表 3-3 の通りである。その他のパラメータについては 3.1.1 節に行われた実験と同じ値を用いる(表 3-2)。学習データ数は 3.1.1 節と同様で擬似データ 10 件とする。ただし、3.1.1 節では 3 種類の平均、3 種類の標準偏差の学習データおよび 3 つのエンコーダにおいて、すべての組み合わせを観測したが、本節では組み合わせは考慮せず、平均 0.25、0.5、0.75 の 3 種類、標準偏差を 0.28 の 1 種類のみ計測を行う。

3.1.4 入力に混入するノイズが不均等だった場合の挙動

これまでの実験では、入力に対するノイズ混入率は 3 つの情報で均等であった。本節では入力に対する混入率が不均等の場合の動作を検証する。対象となるアーキテクチャサイズは 3.1.1 節での実験と同じサイズとする(表 3-1)。その他の最適化パラメータはスパース強度を調整するβのみ変更する。最適化パラメータを表にまとめたものが表 3-4 となる。入力に混入するノイズ率は 3 つのエンコーダに対して、0%、25%、50%の順列からなる 6 種類とすべてが 0%、25%、50%の 3 種類、合計 9 種類で検証を行う。オートエンコーダの初期重みの変え、10 回試行を行う。学習データは平均 0.5、標準偏差 0.28 の正規分布に従う擬似データ 10 件とする。ただし、擬似データは[0.0,1.0]の範囲で生成されている。

表 3-4:最適化パラメータの一覧

$ginitial$	$greach$	$gstep$	α	ρ	β
1.0	10.0	0.5	0.001	0.4	2.0

3.2 結果

3.2.1 モジュール単位での最適化結果

最適化時の評価値の変化：図 3-3 は種類ごとのオートエンコーダの評価値の平均を表している。横軸は誤差逆伝播法の繰り返し回数を表しており、縦軸は評価値であるスパース項が付加された二乗和の値を表している。すべての評価が周

期的に悪化しているのは、オートエンコーダ 3 層目の活性化関数のゲイン値の再設定によるネットワークの微小な変化が原因だと考えられる。

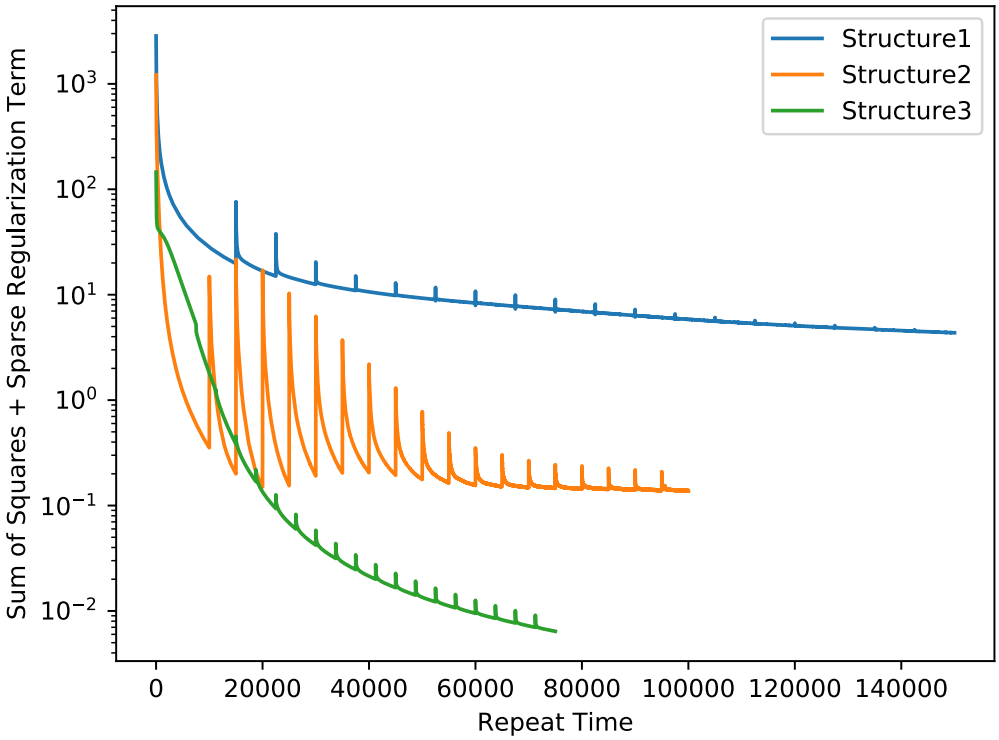

図 3-3:最適化時の評価値の推移

オートエンコーダとしての性能：図 3-4 は情報の種類ごとのオートエンコーダにおける出力と学習データとの誤差を表している。横軸は誤差の混入率を表しており、縦軸は平均二乗平方根誤差を表している。エラーバーは標準偏差を表している。ノイズの混入率が 0%のとき、構造が大きくなるに従い、誤差も大きくなっている。ノイズ混入率が 0 %のときはいずれの構造でも誤差は非常に小さい。ノイズの混入率が 2%のとき誤差は、構造 1 において約 0.23、構造 2 において約 0.2、構造 3 において約 0.07 となっている。ノイズ混入率が 0%のときと比べるとすべての構造において誤差は飛躍的に増加している。ノイズ混入率が 4%以降は緩やかに対数的に誤差が増加する。

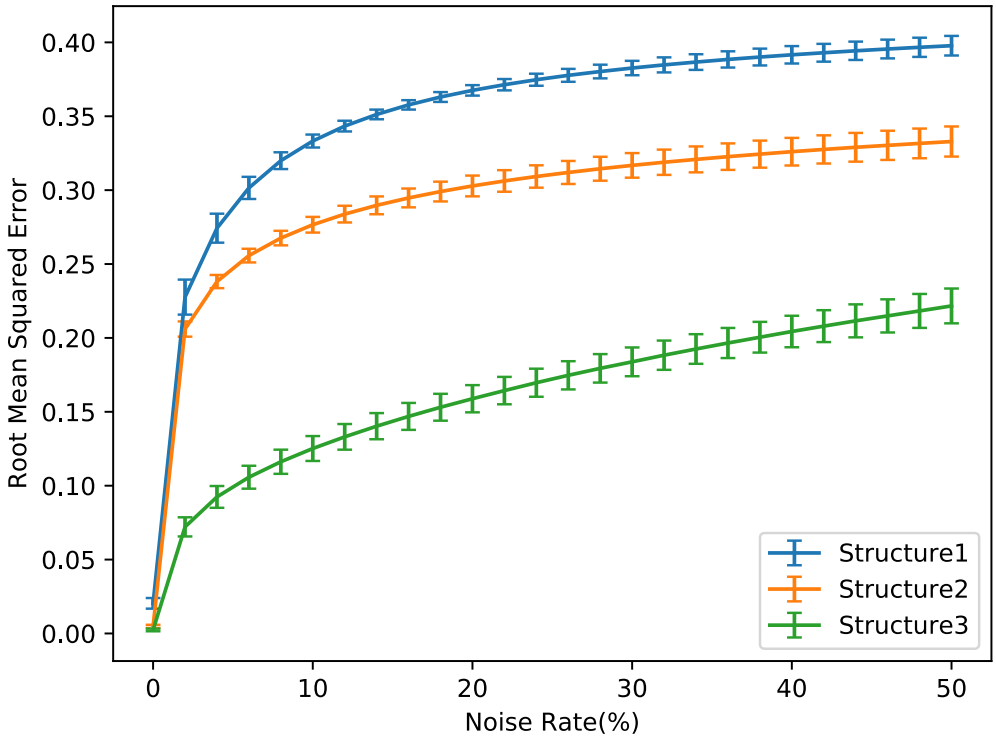

図 3-4:オートエンコーダでのノイズ耐性

アソシエイタでの想起率:図 3-5 はアソシエイタにおける想起率を表している。横軸はエンコーダへの入力に対するノイズ混入率を表しており、横軸は正しいパターンと想起されたパターンのハミング距離を想起率として規格化したものを表している。エラーバーは標準偏差を表している。ノイズ混入率が 0%から 12%までは正しいパターンが想起されている。その後、想起率は徐々に下がって行く。また、想起率が下がるに連れて、標準偏差が大きくなっている。

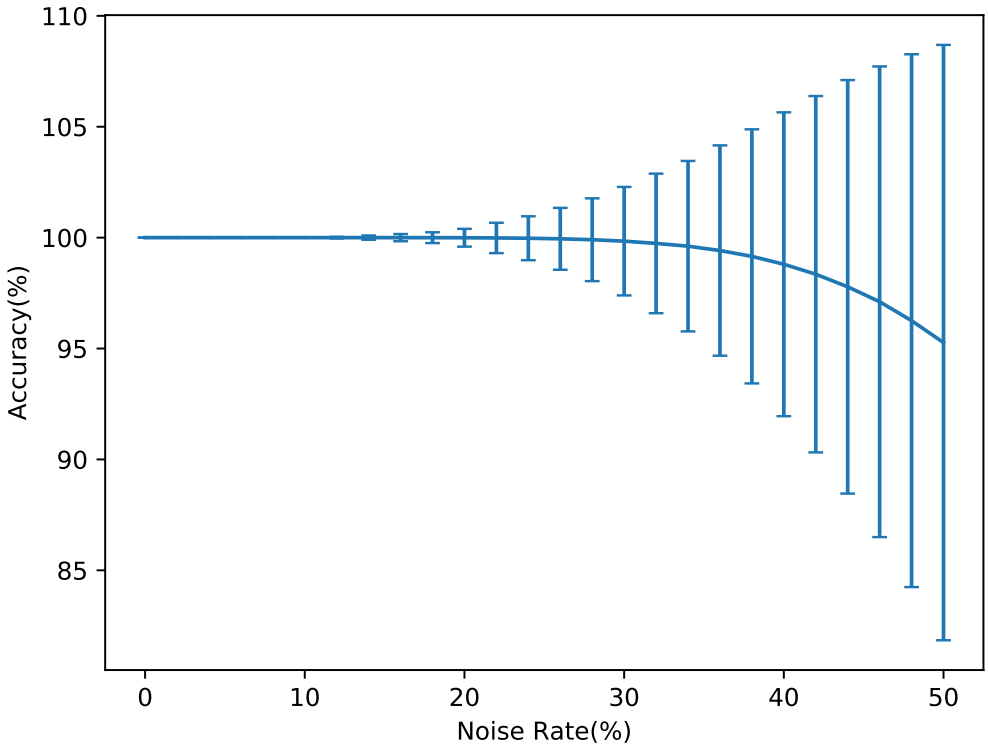

図 3-5:アソシエタでの想起率

正しいパターンが想起されない場合、間違ったパターンを出力することになるが、間違ったパターンを出力する状態は 2 種類あると考えられる。1 つ目は、記憶している別のパターンを出力する場合、もう 1 つは不定状態として銘記されていないパターンを出力する場合である。ノイズ混入率が低く、かつ想起率が 100%ではない時、正しいパターンと比べ距離は近いが、正しいパターンを想起することができない程度には距離が遠いパターンが入力されたと考えられる。これにより、正しいパターンは出力できないまでも、ハミング距離は短いため、想起率も高い状態を維持している。一方、ノイズ混入率が高い時は、正しいパターンと比べ距離が遠いパターンが入力され、正しいパターンとは全く異なるパターンが出力されたと考えられる。ただし、銘記されたパターンと入力されたパターンによっては、正しいパターンとハミング距離が近いパターンが想起されることもあると考えられる。結果としてノイズの混入率が高くなり、想起率が下がっていくにつれて、標準偏差が大きくなったと考えられる。

システムにおけるノイズ除去の性能：図 3-6 はシステムからの出力である各デコーダからの出力と学習データとの誤差を表している。横軸はシステムの入力に対するノイズの混入率を表しており、縦軸はシステムからの出力と学習デー

タの平均二乗平方根誤差を表している。エラーバーは標準偏差を表している。ノイズ混入率が 30%近辺まではノイズを除去できており、非常に低い誤差を維持できている。ノイズ混入率が 30%を超えると誤差は線形に増加する。ノイズ混入率 50%において、最も大きな構造での誤差は約 0.1 であり、3 つの構造の中では最も誤差が大きくなっている。一方、「オートエンコーダとしての性能」で計測された、ノイズ混入率 50%での最も誤差が小さいものは最も小さな構造であり、約 0.2 であった。つまり、ノイズ混入率 50%における提案手法での最大誤差は単純なオートエンコーダでの最小誤差よりも小さい。

ノイズを抑制している範囲において、誤差が一定なのは、アソシエイタで明記されたパターンが正しく想起された結果、デコーダへの入力が一定となったからだと考えられる。

この結果より、単なるオートエンコーダと比べ、アソシエイタを含むシステムは、アソシエイタであるホップフィールドネットワークにおいてノイズ除去することにより、より高いノイズ除去能力を有することを示唆している。

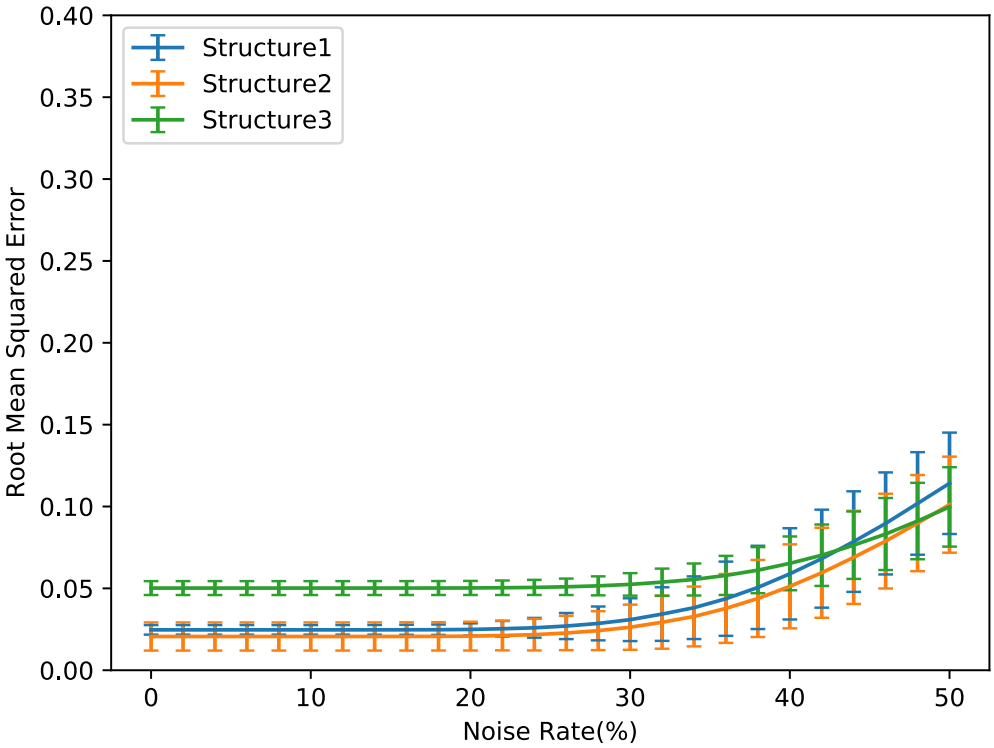

図 3-6:システムとしてのノイズ耐性

3.2.2 提案手法とマルチモーダル情報を処理する単一 Autoencoder の違い

図 3-7 は提案手法とマルチモーダル情報を処理する単一のオートエンコーダとの誤差の違いを表している。横軸は入力に対するノイズ混入率を表しており、縦軸は出力と学習データとの平均二乗平方根誤差を表している。エラーバーは標準偏差を表している。提案手法では前節の「システムにおけるノイズ除去の性能」で示した結果と同様に、30%まで誤差を非常に低く抑えることができている。一方、マルチモーダル情報を処理する単一のオートエンコーダではノイズの混入率に関係なく、誤差は多くなっており、誤差はノイズの混入率の増加に伴い、対数的に増加している。ノイズ混入率が 20%時点においては、提案手法と単一のオートエンコーダでは 6 倍以上もの誤差の違いが発生している。ノイズに対する耐性は提案手法とマルチモーダル情報を処理する単一のオートエンコーダでは明確に差が出ていることがわかる。

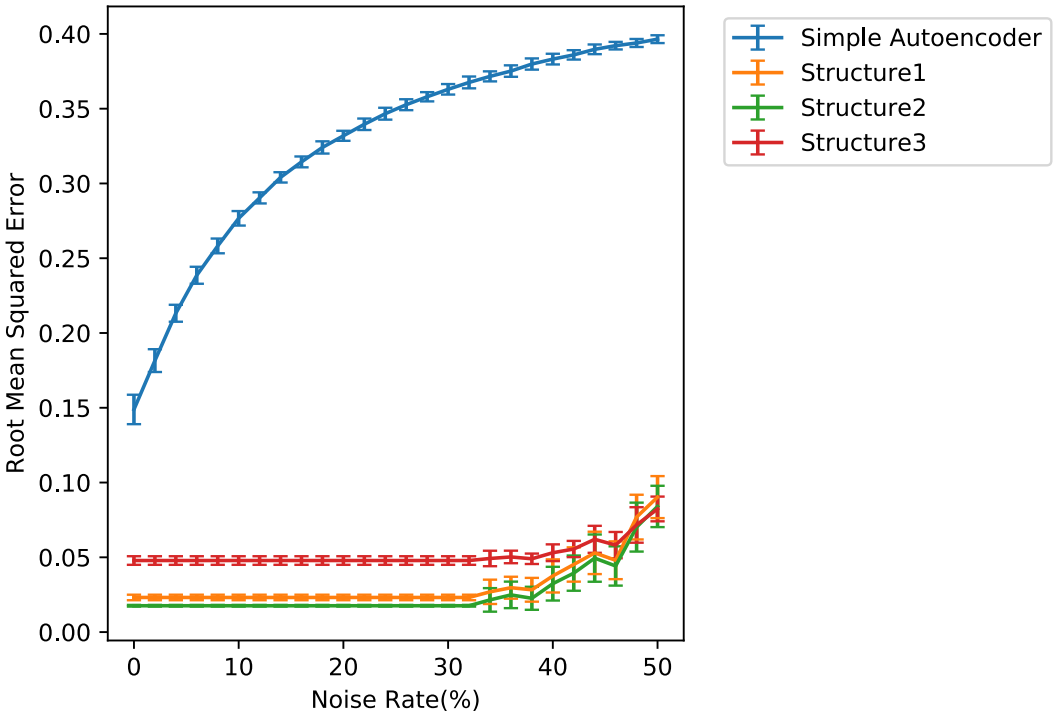

図 3-7:マルチモーダル情報を処理する単一のオートエンコーダと MACMSA のノイズ耐性比較

3.2.3 ネットワークサイズによるノイズ耐性の関係

オートエンコーダとしての性能：図 3-8 はネットワークサイズおよび種類ごとのオートエンコーダにおける出力と学習データとの誤差を表している。横軸は誤差の混入率を表しており、縦軸は平均二乗平方根誤差を表している。エラーバーは標準偏差を表している。ノイズの混入率が 0%の時、誤差はすべてのアーキテクチャサイズにおいて非常に低い。3.2.1 節の「オートエンコーダとしての性能」で示された結果と同様に、入力に 2%のノイズが混入すると誤差は急激に増加し、その後ノイズの混入率の増加とともに対数的に誤差は増加する。ノイズ混入率が 2%で誤差が急激に増え、その後対数的に誤差が増加する傾向は、アーキテクチャサイズに関係なく存在する。

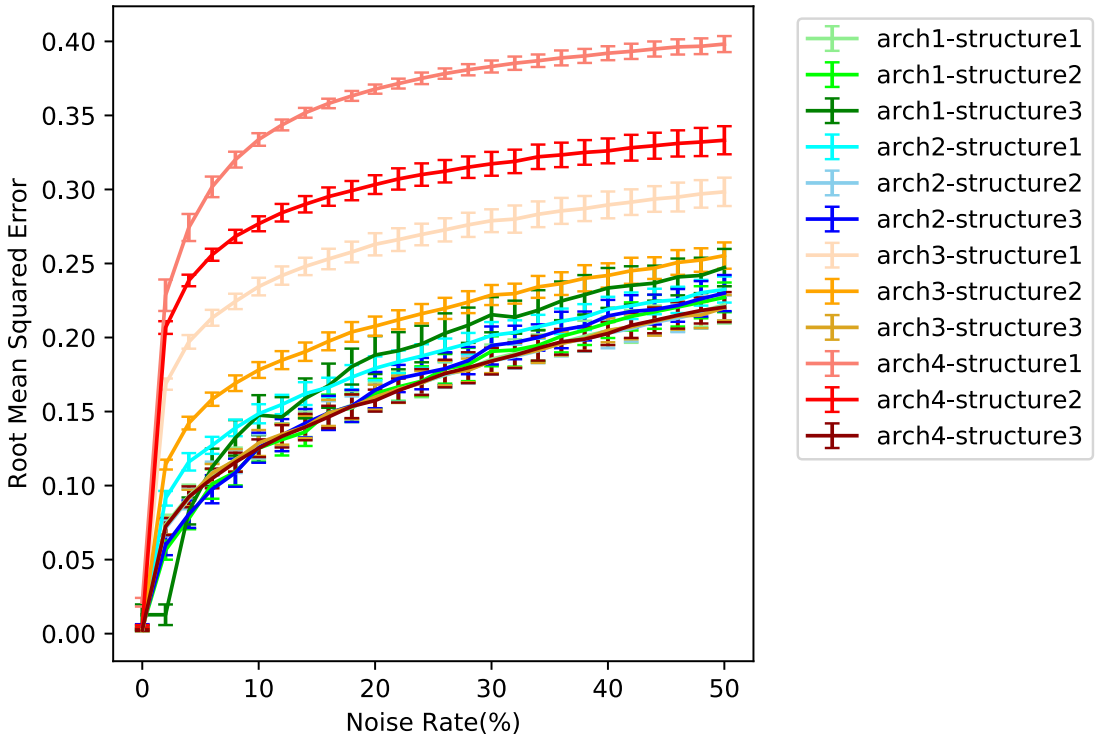

図 3-8:オートエンコーダのネットワークサイズによるノイズ耐性の比較

アソシエイタでの想起率：図 3-9 はアーキテクチャサイズごとのアソシエイタにおける想起率を表している。横軸はエンコーダへの入力に対するノイズ混入率を表しており、横軸は正しいパターンと想起されたパターンのハミング距離を想起率として規格化したものを表している。エラーバーは標準偏差を表している。表 3-5 はアーキテクチャサイズごとのホップフィールドにおける記憶容

量とローディングレートと呼ばれる記憶容量に対し、どの程度パターンを銘記したかを表す値をまとめたものである。記憶容量については式 2-12 におけるスパース度を調整するパラメータである ρ を 0.4 とすることで、スパース化されていないホップフィールドネットワークに比べ、4%の記憶容量向上が行われている。アーキテクチャサイズが最も小さいものを除き、エンコーダへの入力にノイズが混入しても正しいパターンが想起できていることがわかる。ノイズ除去の能力はアソシエイタであるホップフィールドネットワークのサイズに依存し、ネットワークサイズが大きくなるとノイズ除去能力も増加する。

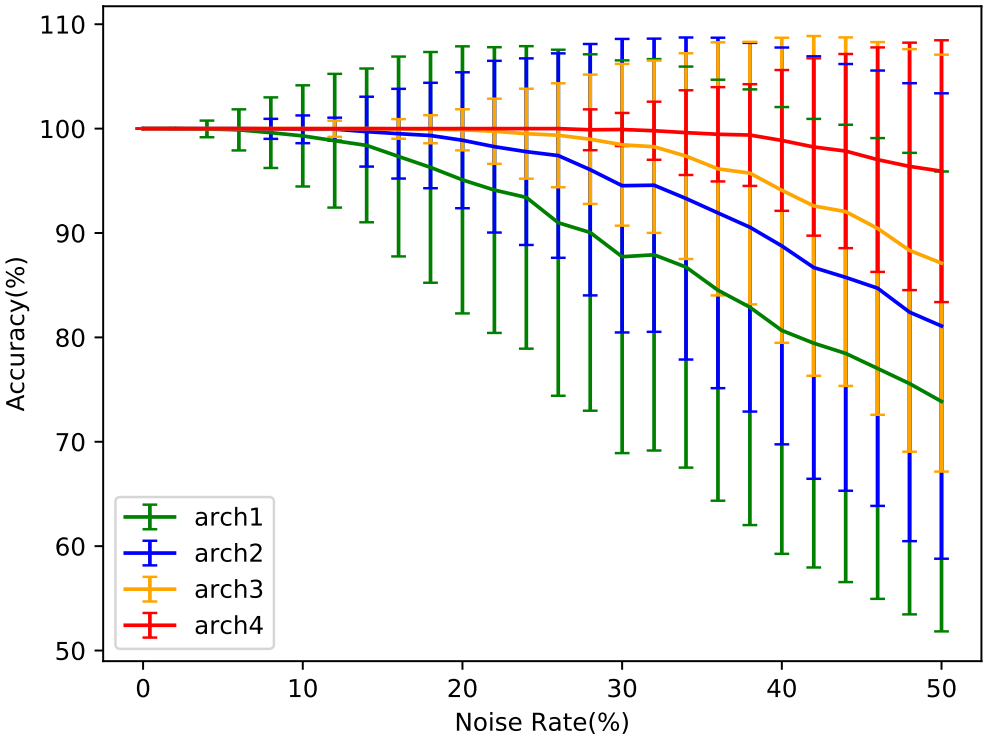

図 3-9:アーキテクチャサイズの違いによるアソシエイタでの想起率の比較

表 3-5：記憶容量とローディングレート

Architecture	a	Storage capacity	Loading rate
Arch 1	0.211	$0.144N$	66.1%
Arch 2	0.206	$0.144N$	38.6%
Arch 3	0.203	$0.144N$	19.4%
Arch 4	0.205	$0.144N$	8.7%

システムにおけるノイズ除去の性能：図 3-10 はシステムからの出力である各デコーダからの出力と学習データとの誤差を表している。横軸はシステムの入

力に対するノイズの混入率を表しており、縦軸はシステムからの出力と学習データの平均二乗平方根誤差を表している。エラーバーは標準偏差を表している。システム全体としてのノイズ耐性はアーキテクチャサイズが大きくなるにつれて増加している。ホップフィールドネットワークにおける記憶容量の観点から述べると、ローディングレートが低くなるにつれて、システム全体のノイズ耐性は高くなる。ホップフィールドネットワークのローディングレートが 10%未満であるアーキテクチャ 4(最も大きなアーキテクチャ)では入力に対するノイズ混入率が 30%近辺までノイズを抑制できている。ノイズを抑制できる範囲はローディングレートの増加とともに狭くなることがわかる。ノイズを抑制できる範囲を超えたノイズが入力された場合、誤差は線形に増える。

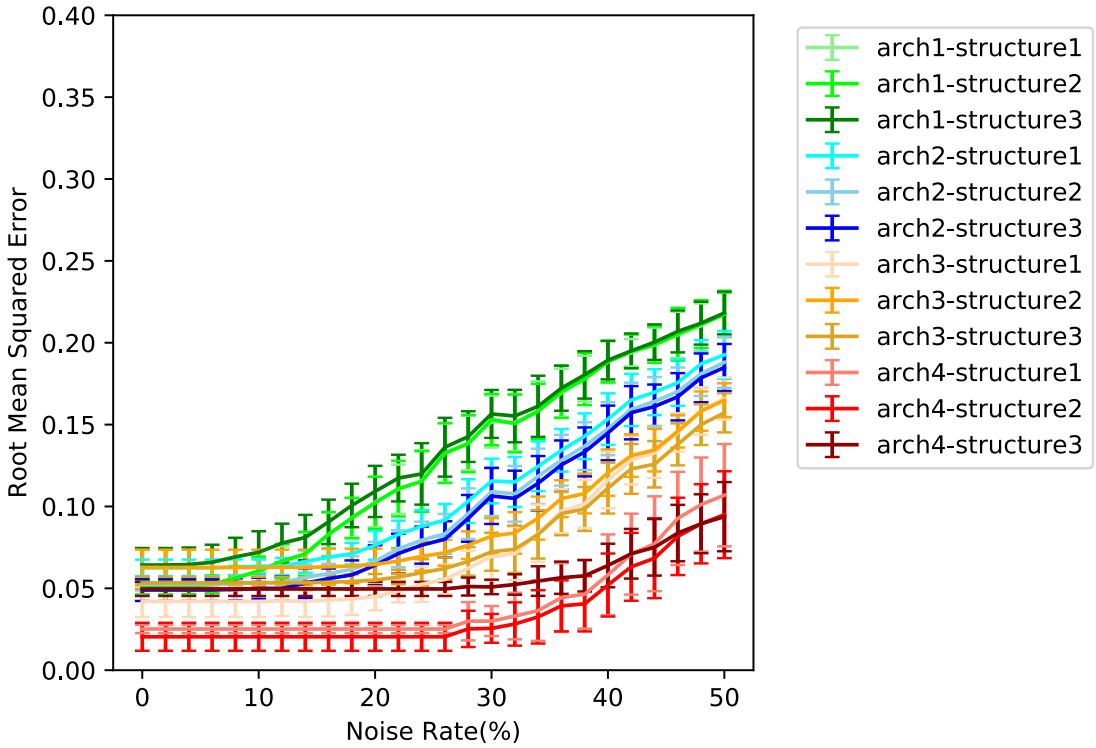

図 3-10:アーキテクチャサイズの違いによるノイズ耐性の比較

3.2.4 入力に混入するノイズが不均等だった場合の挙動

表 3-6 は入力に対するノイズの混入率が 0%、25%、50%の順列およびすべてが 0%、25%、50%における出力と学習データの誤差を一覧でまとめたものである。ノイズ混入率がすべて 0%、25%のときおよび、すべての順列において、平均誤差は、最も大きな構造と 2 番目に大きな構造で 10^{-4} 程度、1 番小さい構造では 10^{-3} 程度となっている。一方、ノイズ混入率がすべて 50%のとき、平均誤差はすべての構造で 10^{-3} 程度となっている。つまり、ノイズ混入率がすべて 50%の時、ほかのノイズ混入率の組み合わせに比べ、1 番大きな構造と 2 番目に大きな構造で 10 倍、一番小さい構造で約 3 倍精度が違うことがわかる。この結果より、システムへの入力に対するノイズが情報の種類ごとに不均等だったとしても、同程度の誤差に抑えることができると言える。また、すべてのノイズが 50%となる組み合わせ以外において、Struct3 では、Struct1 および Struct2 に比べ誤差に 10 倍の差がある。もっとも小さい構造において、誤差が大きくなるのは 3.2.2 節での実験でも同様の結果が示されている。これはネットワークサイズによる最適化精度の限界だと考えられる。

表 3-6:ノイズ混入率が均等でなかった場合の誤差

ノイズ混入率			出力誤差					
Struct1	Struct2	Struct3	Struct1		Struct2		Struct3	
			平均	標準偏差	平均	標準偏差	平均	標準偏差
0%	0%	0%	1.50×10^{-4}	1.51×10^{-4}	3.06×10^{-4}	2.14×10^{-4}	1.30×10^{-3}	1.33×10^{-3}
0%	25%	50%	1.50×10^{-4}	1.51×10^{-4}	3.06×10^{-4}	2.14×10^{-4}	1.30×10^{-3}	1.33×10^{-3}
0%	50%	25%	1.50×10^{-4}	1.51×10^{-4}	3.06×10^{-4}	2.14×10^{-4}	1.30×10^{-3}	1.33×10^{-3}
25%	25%	25%	1.50×10^{-4}	1.51×10^{-4}	3.06×10^{-4}	2.14×10^{-4}	1.30×10^{-3}	1.33×10^{-3}
25%	0%	50%	1.50×10^{-4}	1.51×10^{-4}	3.06×10^{-4}	2.14×10^{-4}	1.30×10^{-3}	1.33×10^{-3}
25%	50%	0%	2.72×10^{-4}	3.69×10^{-4}	4.68×10^{-4}	4.97×10^{-4}	1.72×10^{-3}	1.69×10^{-3}
50%	0%	25%	1.50×10^{-4}	1.51×10^{-4}	3.06×10^{-4}	2.14×10^{-4}	1.30×10^{-3}	1.33×10^{-3}
50%	25%	0%	1.62×10^{-4}	1.54×10^{-4}	3.72×10^{-4}	3.10×10^{-4}	1.46×10^{-3}	1.28×10^{-3}
50%	50%	50%	1.84×10^{-3}	1.51×10^{-3}	2.36×10^{-3}	1.52×10^{-3}	3.76×10^{-3}	2.77×10^{-3}

3.3 考察

本節では 3.2 節で行われた 4 つの実験から示される 6 つの結論を基に MACMSA における有用性を、背景で指摘したコストとノイズ耐性の 2 点の観点より考察を行う。6 つの結論は次の通りである。(i)提案された手法である MACMSA はコンセプト通りの挙動を示す。(ii)エンコーダおよびデコーダは様々な平均、分散を持つデータに対して学習可能である。(iii)アソシエイタであるホップフィールはエンコーダで符号化されたパターンを銘記、想起が可能である。(iv)エンコーダ、アソシエイタ、デコーダで構成されたシステムは、単一のオートエンコーダでマルチモーダル情報を処理するより誤差を小さくすることができる。(v)アーキテクチャサイズによらず、ノイズを含む入力に対し、誤差を低く抑制できる範囲が存在し、その範囲外では誤差は線形に増える。(vi)ノイズが情報の種類ごとに不均一だったとしても、MACMSA は誤差を低く抑えることができる。

コスト：マルチモーダル情報を処理する単一のオートエンコーダの場合、入力層のニューロンは全ての情報の次元を足し合わせたものになる。この点においては提案手法も同様で、各エンコーダの入力層のニューロンの総和はすべての情報の次元を足し合わせたものとなる。しかし、提案手法とマルチモーダル情報を処理するオートエンコーダでは最適化される重みの数に大きな違いがある。3.12 節「提案手法とマルチモーダル情報を処理する単一オートエンコーダの比較」で扱ったネットワークサイズを例としてあげる。提案手法において 1 つ目の隠れ層のニューロン数の和は 1200 ニューロンとなり、ニューロン数はマルチモーダル情報を処理する単一オートエンコーダと同じである。しかし、提案手法では各エンコーダが独立しているため、最適化される重みの数は$1000 \times 750 + 500 \times 375 + 100 \times 75 = 945{,}000$となる。一方、単一のオートエンコーダでは$1600 \times 1200 = 1{,}920{,}000$となる。同様に各層のニューロン数を基に最適化される重みの数を計算すると、提案手法では合計2,853,000の重みが最適化されることになり、単一のオートエンコーダでは合計5,760,000の重みが最適化されることとなる。これによりニューロン数は同数なのに関わらず最適化を行う重みの数は約 2 倍も異なる。ホップフィールドネットワークにパターンを銘記する計算コストについては、オートエンコーダの最適化に比べると遥かに小さく、無視できる程度である。このように独立したエンコーダ、エンコーダの構造をとることで計算コストを削減することができる。また、Centos 7.9、gcc ver.4.8.5、”o3”を最適化オプションとして指定、シングルスレッド使用、Intel(R) Core(TM) i7-3960X、メモリ容量 16GB の条件で、3.1.2 節の実験と同様の条件で、学習にかかる時間の計測を行った。単一のオートエンコーダでは実時間で

10 時間 2 分 43 秒であり、提案手法では実時間で 7 時間 26 分 38 秒であった。最適化する重みが 2 倍であるのにもかかわらず、時間が 2 倍になっていないのは、誤差逆伝播法の繰り返し回数が、提案手法の方が合計で多くなるためだと考えられる。誤差逆伝播法の繰り返し回数の増加を考慮しても、提案手法の方が学習にかかる実時間は短くなっている。

100 次元の情報を 1 つ追加することを考える。隠れ層の数を 3.1.2 節と同じように、第 1 隠れ層のニューロン数を入力層のニューロン数の 0.75 倍、第 2 隠れ層のニューロン数を入力層のニューロン数の 0.5 倍とすると、提案手法では独立したオートエンコーダを学習するだけであるため、75,000 個の重みを追加で学習することになる。一方、単一のオートエンコーダでは 4,040,240,000 個もの重みを追加で学習することになる。つまり、提案手法では情報の種類を追加するために増える学習コストは独立した線形に増えるが、単一のオートエンコーダでは指数的に増える。

本手法は本来、統合すべき情報の種類が多いヒューマノイド型ロボットに適応することを想定して提案されている。しかし、現在においては、モバイルロボットと呼ばれる小型のロボットにおいても複数のセンサーを搭載しているものが多く、容易に複数種類の情報を統合できる本手法はヒューマノイド型ロボットに限らずモバイルロボットなどにも適応可能だと考えられる。

単一のオートエンコーダでもいくつかの結合を省くことで計算コストを削減することができる。しかし、背景で触れたようにどの結合を省けば良いのかは明白ではないため、システムの設計者の大きな負担となる。提案手法では簡素な構造を持つため、結合の省略などを考慮する必要がなく、システムの設計者に対しての負担も小さいと考えられる。

ホップフィールドネットワークの類似手法として、ボルツマンマシンがある [68]。ホップフィールドネットワークは決定論で動作し、ボルツマンマシンは確率論で動作するという、動作の原理に大きな違いがある。ボルツマンマシンの結合に制限を加えることで計算コストを削減した制限付きボルツマンマシンという手法も存在する。この手法は隠れ層を含み、隠れ層の個数を増やすことにより表現能力が向上する。制限付きボルツマンマシンは生成モデルとして取り扱われることもあるが、ノイズ除去能力を有している。そのため、制限付きボルツマンマシンとホップフィールドは交換可能なように思える。しかし、ホップフィールドネットワークのパターン銘記の方が制限付きボルツマンマシンの最適化に比べ計算コストが低く、実装コストも低いと考えられる。したがって、制限付きボルツマンマシンではなく、ホップフィールドネットワークを採用することにはメリットがある。また、制限付きボルツマンマシンはオートエンコーダと類似の働きを持つ。しかし、現在の研究ではネットワークの事前学習に使われること

が多く単独で使われることは多くない[67]。制限付きボルツマンマシンは未だ計算コストの問題はあるが、最適化を高速化する近似方法など研究され続けている。そのため、計算コストの問題が解決すれば、エンコーダ、アソシエイタ、デコーダのすべてのモジュールを制限付きボルツマンマシンに置き換えることを検討する必要があると考えられる。

記憶を取り扱うニューラルネットワークとして、Long Short Term Memory(LSTM)[69]がある。LSTM は以前の入力を記憶する機構を持ち、短期、長期での記憶を可能にする。MACMSA のように単に現状と記憶を比較するという方策を取る場合には LSTM における計算コストは大きなデメリットになり得る。しかし、ロボット制御において、長い一連の行動を制御するなど、これまでの動作履歴を参照し、それに応じて次の行動を決定するという方策を取る場合は有効なように思える。

ノイズ耐性：実環境において、環境ノイズは光源や音源、その他様々な要因で頻繁に混入すると考えられる。環境ノイズが存在しなかったとしても、アナログ回路を有したセンサーを使えば、全く同じ入力を得ることは困難である。本研究では、学習データと実環境での齟齬を考慮し、システムの入力に対して最大 50% のデータが書き変わる実験を行った。3.1.1 節の「システムにおけるノイズ除去の性能」で示したように、ホップフィールドネットワークのローディングレートを低くすると、誤差を極めて低く抑えることができる範囲を伸ばすことができる。また、その範囲を超えたあとは、線形に誤差が増える。このノイズ除去能力は、アソシエイタに由来すると考えられる。エンコーダまたはデコーダが強いノイズ耐性を有していたと仮定すると、3.1.1 節の「オートエンコーダとしての性能」の結果の結果に反する。したがって、オートエンコーダがアルゴリズム上強いノイズ耐性を有しているとは言い難い。ただし、3.1.1 節の「システムにおけるノイズ除去の性能」の結果より、デコーダだけに着目すると、デコーダへの入力となるアソシエイタからの出力が銘記されたパターンに近ければ、学習データに近い値を出力することができる。ノイズ混入率が 50%の時点に着目すると、アーキテクチャサイズが大きくなるにつれ、MACMSA と単一のオートエンコーダのそれぞれの誤差の差は大きくなる。

実環境においては、センサーから得られる情報の次元は本実験で扱ったものよりも大きくなると考えられる。例えば、カメラを用いて、64×64pixel の画像を得えたとすると、エンコーダおよびデコーダとなるオートエンコーダは 3.1.1 節で用いた最も大きなオートエンコーダに比べ約 4 倍も大きくなる。これにより、アソシエイタのネットワークサイズも大きくなり、記憶容量も向上し、MACMSA としてのノイズ耐性も向上すると考えられる。さらに、スパース強度を調整するパラメータである ρ を本研究で用いた値より小さくすることで銘記

するパターンはさらにスパース化され、記憶容量の向上を計ることができる。ただし、スパース強度を高めることにより、ノイズ耐性の低下が考えられるため、極端に高いスパース強度の設定は避けるべきだと考えられる。

ホップフィールドネットワークに着目すると、ノイズ耐性は理論値より低い。理論研究では明記されるパターンは理想的なパターンであると仮定される。一方、エンコーダから生成されるパターンは理想的なパターンとなっていない。このことが、理論値よりノイズ耐性を低下させている原因だと考えられる。エンコーダで生成されるパターンが線形独立となっているなど理想パターンを生成できるようになればノイズ耐性も理論値に近い値になると考えられる。

本研究では最大 50%ノイズ混入率があると考え実験を行なった。しかし、実環境では同一状況で 50%もノイズが混入するとは考えにくい。したがって、ある学習データと 50%以上相違があるデータが存在した場合、そのデータは新規の状況であると考え、学習データに加え学習し直すことが適切であると考えられる。

MACMSA の欠点を挙げるとするならば、ロボット制御における最も重要な機能が 1 つ欠けていることである。この機能とは、現在の状況から制御用の信号を生成する機能である。この機能については次章において、MACMSA を改良することで追加することとする。

最後に、3.2.1 節から 3.2.3 節までで示した結果は、C 言語を用い、1 から構築されたプログラムで得られた結果である。一方、3.2.4 節で示した結果は python における深層学習構築用ライブラリである PyTorch[70]を用いて構築されたプログラムで得られた結果である。PyTorch などのライブラリを用いることで、簡単に深層学習のシステムを構築することができる[71]。しかし、PyTorch を用いて実験を行なった場合、C 言語を用いて書かれたプログラムで使用された最適化に関するパラメータでは、同一結果を得ることができなかった。C 言語で構築されたプログラムでは損失関数の微分値は直接離散値として求めるのに対し、PyTorch では自動微分を使った近似値が適応されるからだと考えられる。ただし、スパース項を調整することで、C 言語で構築されたプログラムでも、PyTorch で構築されたプログラムでも同様の結果となることが確認されている。

3.4 まとめ

本章では MACMSA の基本性能を解析した結果を示した。MACMSA は様々な平均、標準偏差を持つデータを学習することができ、かつ高いノイズ混入率でも安定して、誤差の少ない出力を出すことができる。ノイズ混入率が情報の種類によって不均等な場合でも同様に安定した出力を得ることができる。このノイズ

耐性はアーキテクチャのサイズに依存し、アーキテクチャが大きくなればノイズ耐性も高くなる。

第 4 章　提案手法の改良と実データにおける動作検証

4.1 導入

前章において提案された手法にはロボット制御システムとして重要な機能が 1 つ欠損している。本章では、ロボット制御システムとして MACMSA を稼働させるために手法の改良を行う。検証実験として、リーチングタスクを行うロボットアームから得られた実データを用いて学習実験を行う。システムの最適化完了後、学習に用いた実データを最適化されたシステムに再度入力し、このとき生成された出力を実際のロボットアームで実行する。この実験により、改良された提案手法が実データを学習できることを示し、特定の実データが入力された際に生成される制御信号が実ロボット制御においてどのように実行されるかを検証する。

4.2 手法の改良と最適化方法

前章で提案された MACMSA と、本節で改良を行う手法で最も異なる点はアソシエイタであるホップフィールドのニューロン数である。改良された手法では、1 つのホップフィールドネットワークで時刻 t と時刻 $t+1$ の状態を同時に銘記および想起を行う。そのため、MACMSA に比べニューロン数の多いホップフィールドネットワークが採用されている。改良された手法の概念図は図 4-1 の通りである。

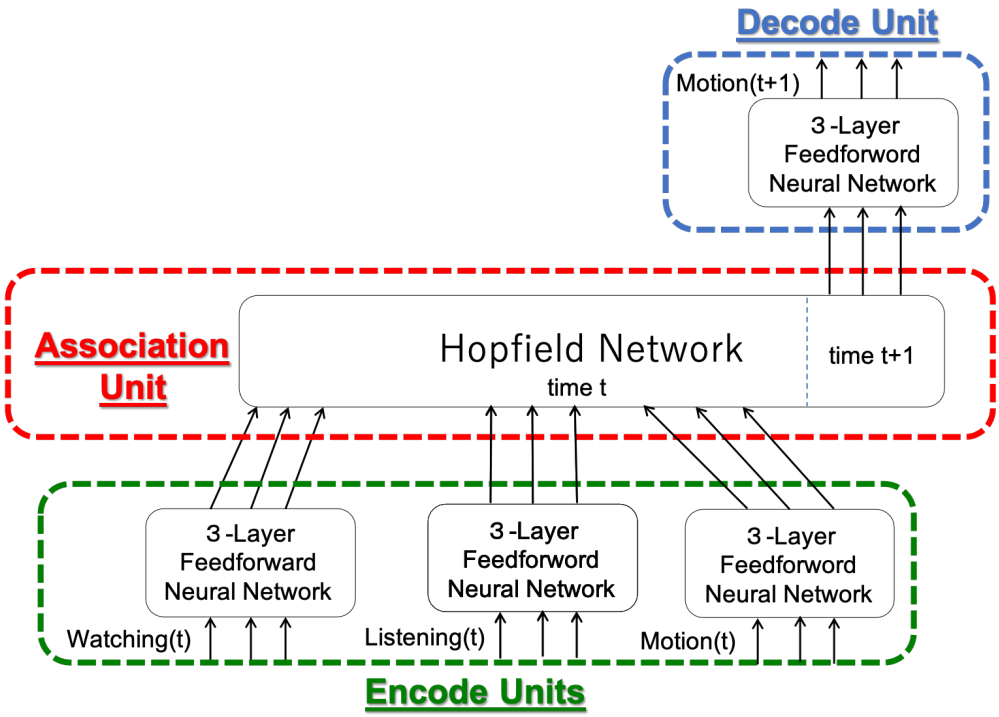

図 4-1:改良された MACMSA の概念図

改良された手法は MACMSA と同様に、エンコーダユニット、アソシエイタユニット、デコーダユニットで構成されている。エンコーダユニットでは情報の種類ごとに 3 層のフィードフォワードネットワークで構成されるエンコーダを有する。エンコーダはセンサーから得た時刻 t の状態をアソシエイタで処理できる 2 値へと変換を行う。アソシエイタユニットは 1 つのホップフィールドネットワークで構成されており、2 値に変換された時刻 t の状態と、時刻 $t+1$ の状態である制御信号が銘記および想起される。デコーダユニットはアソシエイタで想起された時刻 $t+1$ の状態を、センサーへ入力できる実数値へと変換を行う。デコーダユニットは MACMSA と違い、アクチュエータと繋がるデコーダのみが存在する。

システムの最適化方法は MACMSA と同様の 3 ステップに加え、デコーダのチューニングを 4 ステップ目として行う。各ステップについては次節にて説明を行う。

4.2.1 エンコーダとデコーダの最適化

MACMSA ではすべての活性化関数がシグモイド関数であったが、改良された手法ではエンコーダ 3 層目の活性化関数はハイパーボリックタンジェント(式 4-1)が採用される。MACMSA ではエンコーダで符号化された情報は近似的な 0 と 1 であり、アソシエイタにおいて銘記時、想起時ともに{−1,1}へ置き換えが行われ

ていた。しかし、エンコーダ 3 層目の活性化関数を、値域が[−1,1]であるハイパーボリックタンジェントとすることで、想起される際の置き換えを無くすことができる。ただし、銘記時は 0.0 を閾値に{−1,1}の置き換えは行われる。

$$Tanh(\boldsymbol{x}) = \frac{e^{\alpha x} - e^{-\alpha x}}{e^{\alpha x} + e^{-\alpha x}} \tag{4-1}$$

エンコーダとデコーダの最適化に関する手順は 2.2.2 節で述べた MACMSA の最適化手順(図 2-4)と同じで、式 4-1 における α の値をゲイン値とし、最適化の過程で徐々に高く設定を行い、学習完了時において、エンコーダからの出力が近似的に−1 と 1 になるように最適化を行う。最適化手法は Adam[72]へと変更する。

4.2.2 アソシエイタに銘記するパターンの生成とアソシエイタへの銘記

全学習データを各情報のエンコーダを用いて近似された−1 と 1 に置き換える。その後、閾値を 0.0 として−1 および 1 に置き換えを行う(図 4-2)。

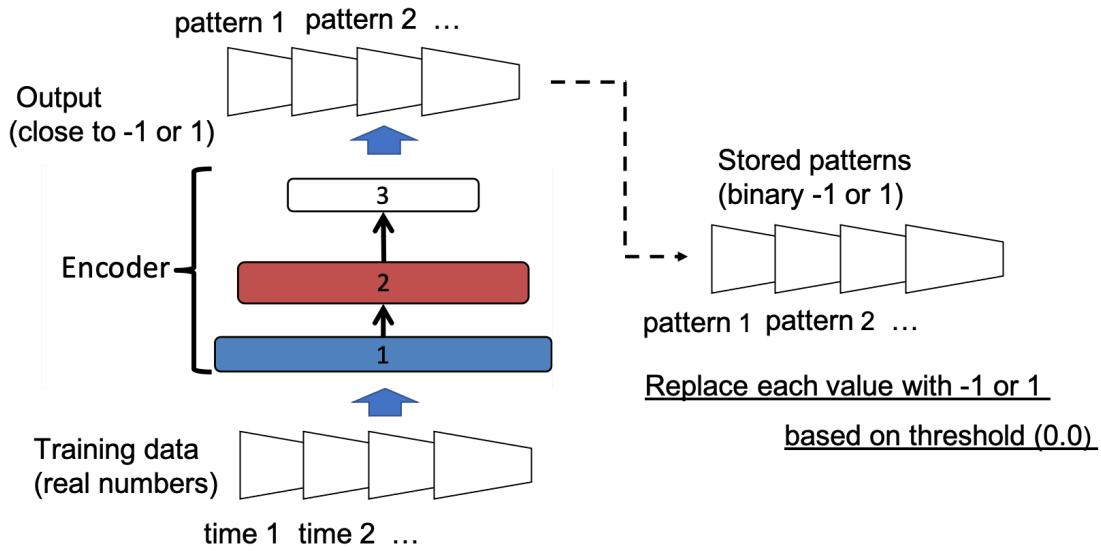

図 4-2:近似的−1 と 1 から完全−1 と 1 への置き換え

この置き換えはエンコーダからの出力が近似的に−1 と 1 であるが、正確に 2 値になっていないため行われる。符号化された 3 種類の学習データは、時系列ごとに結合される。また、時刻 *t+1* の状態についてもこの段階で結合される。ロ

ボットの制御信号を出力する部分にはアクチュエータの位置情報の時刻$t+1$を結合する。つまり、画像用のエンコーダから符号化されたパターンを$W[i]$、音声用エンコーダから符号化されたパターンを$L[i]$、アクチュエータの位置情報用のエンコーダから符号化されたパターンを$M[i]$とすると、ホップフィールドネットワークに銘記するパターンは$W[i]$、$L[i]$、$M[i]$、$M[i+1]$を結合したものとする。ただし、iは時系列に沿ったデータ番号であり、$i=\{0,1,\dots,N\}$とする。Nはデータ件数−1である。また、画像、音声、アクチュエータの位置情報のデータ番号がNのときは$W[N]$、$L[N]$、$M[N]$、$M[N]$を結合することとする。

アソシエイタへのパターンの銘記は 2.2.2.2 節で述べた手順と同じであり、{−1,1}へ置き換えられた全情報は結合された状態で、式 2-8 を用いて銘記される。また、想起の方法も 2.2.2.2 節で述べた手順と同じであり、式 2-9 と式 2-10 を用いて想起される。

4.2.3 デコーダのチューニング

デコーダは位置情報を学習したオートエンコーダの中間層から出力層までの 3 層となる。オードエンコーダ学習時、エンコーダからの出力となる中間層の出力は実数として最適化される。一方、推論時、アソシエイタからの想起結果は 2 値である。そのため、誤差が生じる。この誤差を小さくするために、最適化済みオートエンコーダの中間層から出力層までの 3 層を初期状態とし、デコーダのチューニングを行う。学習データは{−1,1}で構成されるパターンとし、教師信号は時刻が一致するアクチュエータの位置情報とする。学習において、損失関数は二乗和とし、誤差逆伝播法の繰り返し回数はアクチュエータの位置情報を学習したオートエンコーダにおけるハイパーパラメータである *bp_loop1* 回とする。

4.2.4 推論時のデータフロー

推論時は以下の手順に従い、システムは制御信号を出力する。まず、モーダルごとの入力に対しエンコーダを使い[−1,1]のパターンに符号化する。次に、すべてのモーダルのパターンを結合する。このとき、時刻 *t+1* に相当するパターンはすべて 0 として結合を行う。結合したパターンをアソシエイタの初期状態として与える。アソシエイタは与えられた初期状態から時刻 *t+1* を含め、想起を行う。想起された時刻 *t+1* の部分を切り取り、デコーダで[−1,1]から元の値域へ変換を行う。データフローを図示したものが図 4-3 となる。

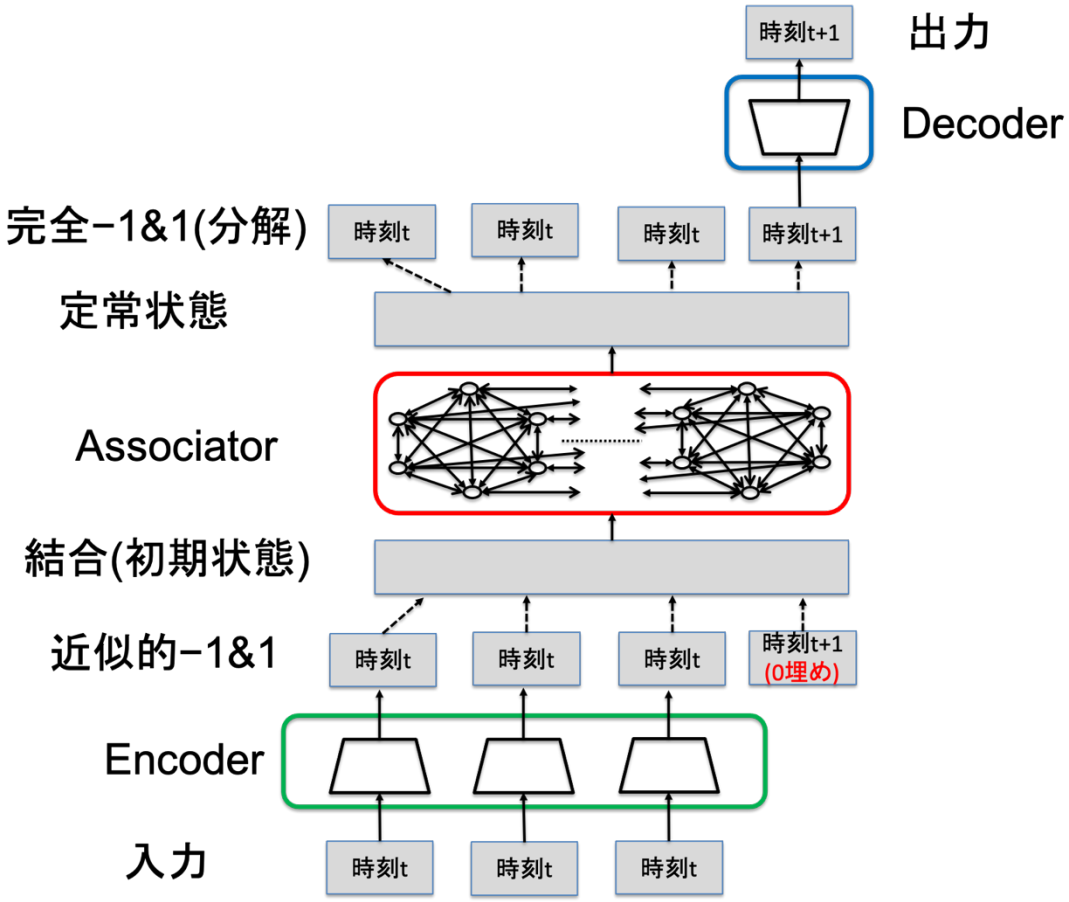

図 4-3:改良された MACMSA の推論時におけるデータフロー

4.3 実データを用いた学習実験

4.3.1 実験装置

本実験では、画像、音声、アクチュエータからの位置情報の 3 つで構成されるデータを用いる。画像はサンワサプライ製の Web カメラである 400-CAM083 を使う[73]。音声は Web カメラ内蔵のマイクを使う。アクチュエータは KXR-A5 アーム型 Ver.2 を使う[74]。この KXR-A5 は 5 つのサーボモータをアーム型に連結したものである。図 4-4 は実験装置の全体像を示したものである。

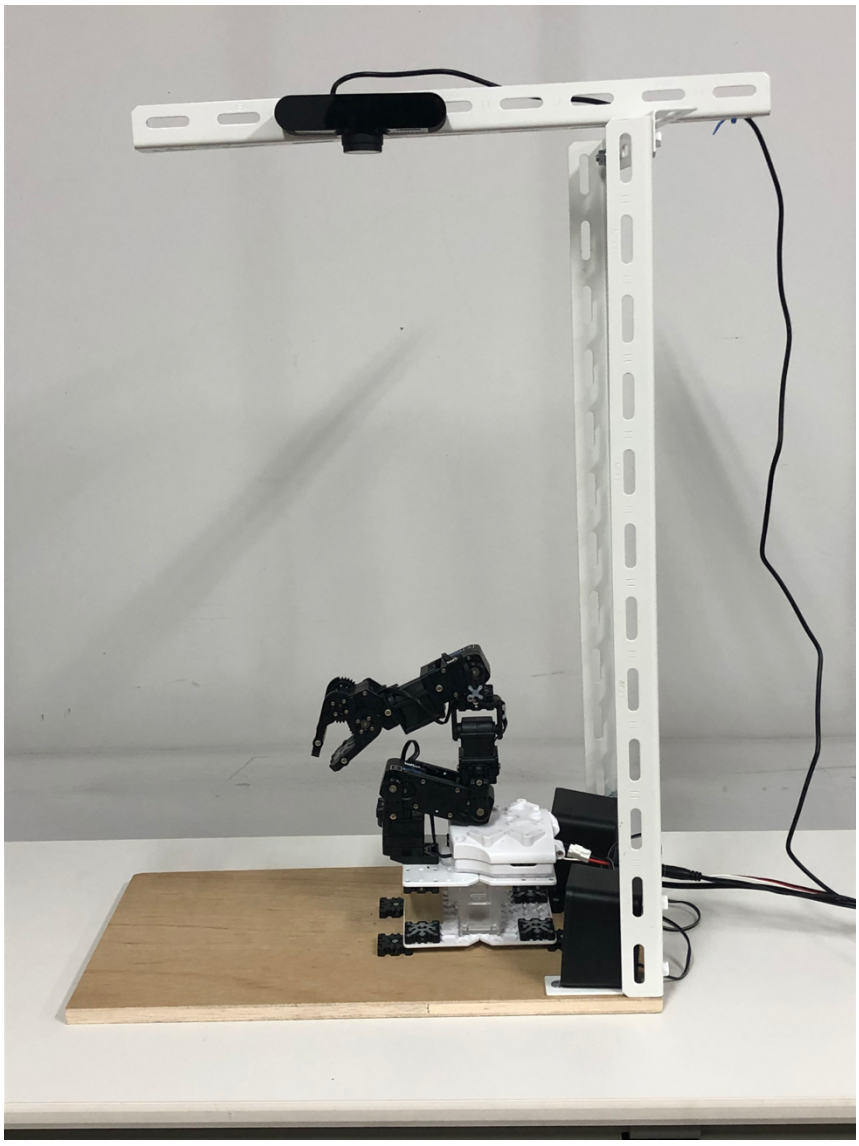

図 4-4:ロボットアームと上部に取り付けられた Web カメラ

ロボットアームは初期状態から中央、右、左方向に対しそれぞれ 2 点を経由しアームを伸ばす。アクチュエータからの位置情報は 5 つすべてのサーボモータからその時点での位置情報を取得する。位置情報は[3500,10500]の整数値で定義される。画像は画像全体からロボットアーム全体が写るように、600×750pixel を切り出す。その後 8bit グレースケール化を行い、20×25pixel へ縮小を行う。図 4-5 は学習データとなるすべての画像を表したものである。

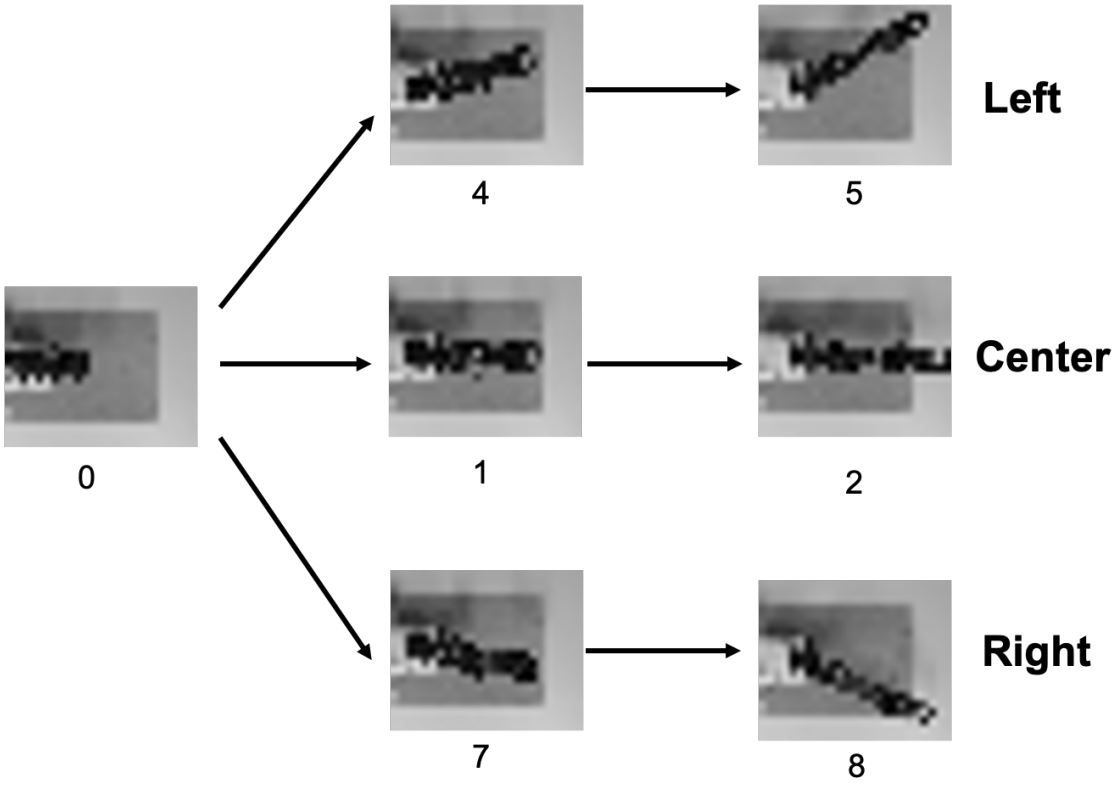

図 4-5:学習される状態の一覧

オートエンコーダ最適化時、画像とアクチュエータの位置情報の 9 データのうち初期状態となる 3 データは同じ状況であるため、1 つにまとめ、7 データとして最適化を行う。音声は、シンバル、リングベル、スネアドラム、フィンガースタイルベース、マリンバ、ハープ、チャーチオルガン、琴、トランペットの 9 音を再生したものを、マイクで録音し、その後フーリエ変換を行ったものが学習データとなる。録音時間は 1 秒、量子化ビットを 16bit、周波数 16kHz とする。フーリエ変換は最初の約 0.08 秒を削除し、次の 500 点をサンプリングする。ただし、各モーダルは、次元に対し、式 4-2 による規格化を行い、値域を[0,1]へ変換されることに注意されたい。

$$\boldsymbol{x}' = \frac{\boldsymbol{x} - \min(\boldsymbol{x})}{max(\boldsymbol{x} - \min(\boldsymbol{x}))} \tag{4-2}$$

ネットワーク構造に関するパラメータは表 4-1 の通りとなる。

表 4-1:アーキテクチャに関するパラメータ

Autoencoder structure	Input layer	Hidden layer 1	Hidden layer 2	Hidden layer 3	Output layer	***bp_loop1***	***bp_loop2***
Picture	500	375	250	375	500	10 000	5 000
Audio	500	375	250	375	250	10 000	5,000
Position	5	10	20	10	5	2 000	1 500

3 章の実験では、すべてのエンコーダにおいて、入力層から出力層に向けて、次元圧縮を行うようにニューロン数を減らしていた。しかし、本研究ではアクチュエータからの位置情報を学習するエンコーダのみ入力層から出力層に向けて次元を拡張するようにニューロンを増やし、最適化を行う。これはアクチュエータからの位置情報は 5 次元しかなく、これ以上次元を圧縮すると 7 件のデータを符号化するのに十分なニューロン数を得ることができないからである。アソ

シエイタに銘記するパターンの結合番号を一覧として表したものが表 4-2 となる。表中の番号はデータ番号であり、画像とアクチュエータの位置については、図 4-5 中の番号と一致する。

表 4-2:結合されるデータ番号の一覧

	画像	音声	アクチュエータ位置	出力(アクチュエータ位置)
パターン 1	0	0	0	1
パターン 2	1	1	1	2
パターン 3	2	2	2	2
パターン 4	0	3	0	4
パターン 5	4	4	4	5
パターン 6	5	5	5	6
パターン 7	0	6	0	7
パターン 8	7	7	7	8
パターン 9	8	8	8	8

最適化に関するその他のパラメータは表 4-3 の通りである。

表 4-3:最適化に関するパラメータ一覧

$ginitial$	$greach$	$gstep$	α		ρ	β
			Picture & Audio	Position		
1.0	10.0	1.0	0.00001	0.005	0.4	1.8

システムの最適化後、学習データをシステムに再度入力し、この時得た出力をロボットアームで実行し、適切に学習できたか検証を行う。このとき、式 4-2 とは逆の手順で[0,1]の値からアクチュエーターの定義域へ変換を行う。

本実験では PyTorch ver.1.8.1 を用いてオートエンコーダの実装と最適化を行った。ホップフィールは PyTorch を用いずに実装した。実装の違いにより 3 章で行った実験とはβの値が異なるが、同等の結果が出るように調整してある。

4.4 結果

学習データをロボットアーム上で実行した時の軌跡と、学習データを最適化済みのシステムに再度入力し、得られた制御信号をロボット上で実行した時の軌跡を比較したものが図 4-6 になる。ただし、システムから得られた制御信号を

ロボット上で再生した軌跡は、オートエンコーダの初期重みを変え 100 回試行した中で、アソシエイタにおける想起率が最も高かった結果である。

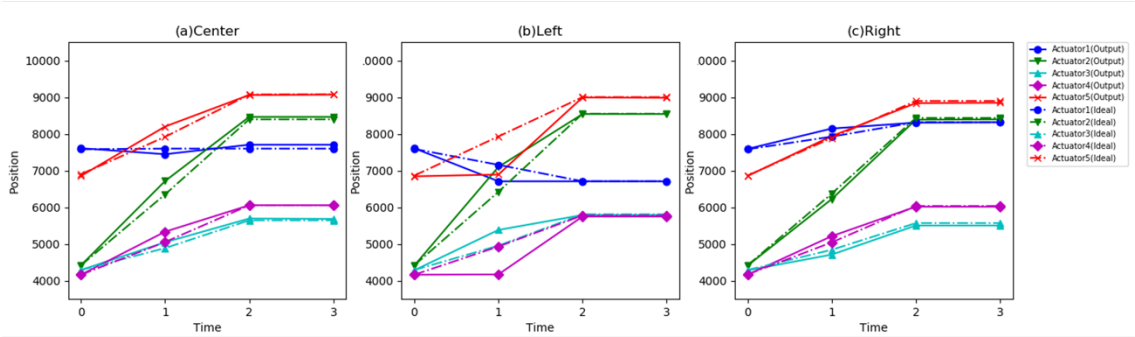

図 4-6:各アクチュエータの軌跡

縦軸がアクチュエータの位置を示しており、横軸が時刻を表している。学習データをロボットアーム上で再生した時の軌跡は破線で表されている。システムから得られた制御信号をロボット上で実行した軌跡は実線で表されている。システムから得られた出力をロボット上で再生した軌跡うち、時刻 1 から時刻 3 までが実際にシステムが出力した制御信号となる。Left の Time1 においてアクチュエータ 5 とアクチュエータ 4 で 1000 程度違いが出ている。また、この結果において、学習データをシステムに入力した時のアソシエイタにおける想起率は全体で約 98.3%、出力に直接影響する部分で約 98.9%であった。つまり、学習データを入力しても完全想起はできなかったことになる。ロボットアーム実機上での動きを目視により観察した結果(図 4-7)、Left においてアームが一部伸びきっていないことがわかった。これはグラフにおける Left のアクチュエータ 4 であった。そのほかにおいては軌跡で確認した通り、学習データをロボットアーム上で実行した場合と、システムから得られた出力をロボット上で実行した場合とで、大きな違いは観察されなかった。

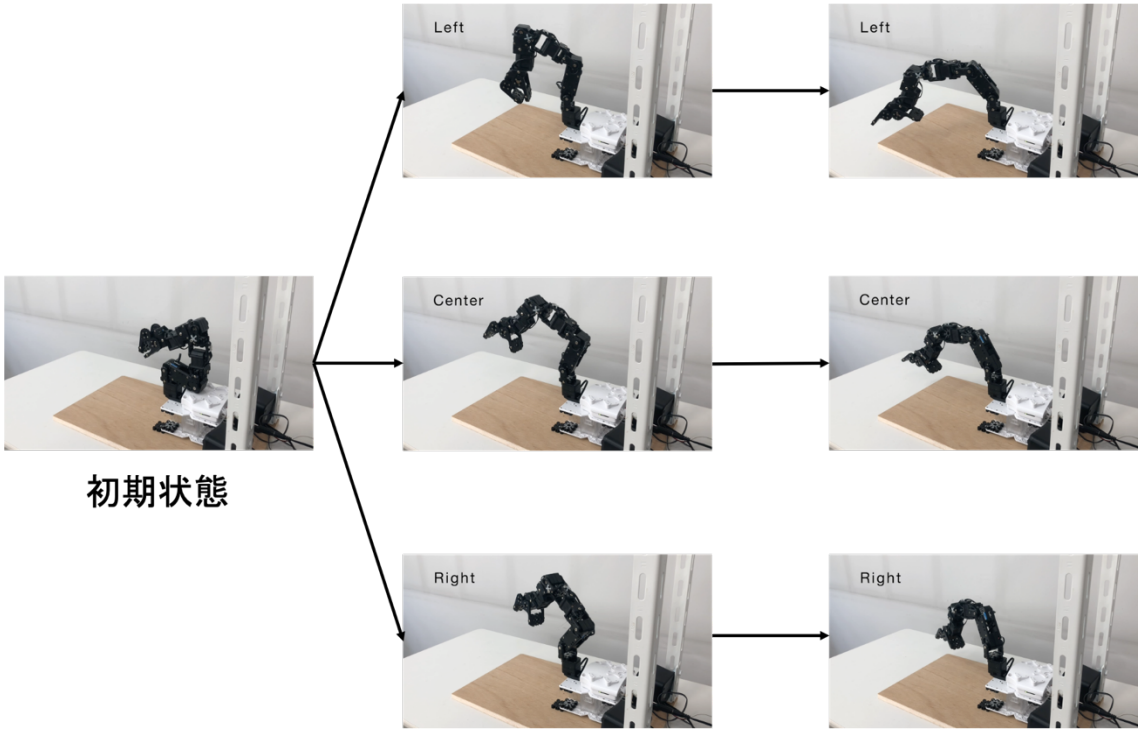

図 4-7:各状態の写真

4.5 考察

最終的な目的位置では、中央、右、左方向すべてにおいて、学習データにしっかりと追従できている。経由点においては、Left の Time1 のときだけ大きな誤差が生じていた。実ロボットアームを制御するという観点においては、制御不能に陥ることはなかったため、問題ない範囲の誤差と考えられる。そもそも、アクチュエータから取得される位置情報はアナログ回路を経由したデータであるため、仮に理想データがセンサーへ入力されたとしても、理想データと全く同じ位置へアクチュエータを動かすことは不可能である。

Left の Time1 で大きな誤差が発生した理由はアソシエイタにおける想起率にあると考えられる。想起率に着目すると、実験では学習データを入力したとしても 100%想起ができなかった。想起率は明記するパターンを生成するエンコーダの能力に大きく依存する。本実験では各センサーで得られた値は簡単な規格化で[0,1]へ変換されている。しかし、オートエンコーダの学習においてデータの白色化[67]を行うことで、より良い精度を出す場合があることが知られているため、データの白色化を行うことで想起率が向上する可能性がある。

本実験では全 9 状態を学習し、初期状態から最大 2 ステップ先の状態までを学習を行った。ロボット制御において、制御する時間間隔を短くし、精密な制御

を行いたい場合も存在すると考えられる。改良された提案手法では現在の状態のみに着目し、次の状態への制御信号を想起する仕組みを採用している。したがって、時間間隔を短く設定し、学習する状態を増やしたとしても、アソシエイタであるホップフィールネットワークにおける記憶容量を上回らない限りは正しく制御信号を想起できると考えられる。ただし、状態間の違いが限りなく小さい場合は、明記するパターンの距離が短くなりすぎ、正しく銘記できない可能性があると考えられる。

MACMSA だけでは時刻 *t* の状態を入力し、時刻 *t* を出力するため、ノイズを考慮しない状況では若干の学習誤差が発生するだけで、オートエンコーダと機能の点では違いはない。一方、本章では MACMSA を基に時刻 *t* と時刻 *t+1* の状態を銘記、想起するようにホップフィールドネットワークのサイズを大きくした。これはホップフィールドネットワークをモジュールとして持っているからこそできる拡張である。オートエンコーダでは時刻 *t* から時刻 *t+1* の状態を生成する機構を持ち合わせていないため、ノイズの有無にかかわらず、オートエンコーダ単体では本章で行った実験を追試することはできない。

リーチングタスクのみを対象とすると、アクチュエータの位置情報のみを入力し、制御信号を生成するという、ユニモーダル情報を扱ったシステムを考えることができる。しかし、本手法はリーチングタスクに限らず、人間との相互作用が生まれる状況におけるロボット制御を想定して設計されている。人間と相互作用する状況ではアクチュエータの位置情報だけでは状況認識が不十分だと考えられる。具体的には人間がどこに位置しているか、何か音を発しているかなど、視覚および聴覚なども並行して取得し処理する必要がある。そのため、人間と相互作用する状況でロボットの制御を行うためにはマルチモーダル情報の処理が不可欠である。また、視覚と聴覚となる装置をロボットに搭載する必要がある。本実験では Web カメラが視覚であり、マイクが聴覚となる。

本実験ではリアルタイムの制御を行わなかった。これはハードウェアの性能上、全く同じ状況であるのにも関わらず得られたデータ間で 20%以上の違いが発生したためである。3 章における結果を参考にすると、本実験に用いられたサイズのアーキテクチャであれば、20%以下の環境ノイズを抑制できると考えられる。改良された MACMSA を用いてリアルタイムでの制御を行うためには、ハードウェアの選定をしっかりする必要がある。特にカメラとマイクの選定が重要である。アクチュエータに関しては同一状況の場合、データ間の違いは 0.2%程度であったため、本実験に用いたロボットロボットアームは採用可能であると考えられる。アーキテクチャを構成するモジュールに関しては、オートエンコーダの層構造の変更だけでは不十分だと考えられ、センサーごとに適したエンコーダおよびデコーダを選ぶとことを検討する必要がある。オートエンコーダの層

構造を変更した場合、学習される特徴量に違いは生まれるが、ノイズ耐性の向上には効果がほぼないと考えられるためである。エンコーダおよびデコーダの候補として、画像であれば Convolutional Neural Network(CNN) [75-77]、音声であれば LSTM [69,78,79]が候補に上げられる。このように情報の種類に適したエンコーダおよびデコーダを用いることでリアルタイムに制御できる可能性がある。アクチュエータの位置情報に関しては、データ間の違いが小さかったため、オートエンコーダで十分であると考えられる。これらアーキテクチャのモジュール変更は実際に使うハードウェアに対するカスタマイズとして行うことが良いだろう。

4.6 まとめ

改良された MACMSA はロボットアームから得た実データを学習することが出来る。また、学習データを最適化済みシステムに再度入力した際に得られる制御信号は実ロボットアーム上で実行可能であり、リーチングタスクをこなすことができる。ただし、アソシエイタにおける想起率は学習データを入力しても100%ではなく、リアルタイム制御も不可能であった。

第 5 章：総合討論

5.1 結果のまとめ

3 章の結果から、MACMSA は様々なデータを学習することができることが示された。また、最大 30%程度までのノイズであればシステムがノイズを抑制できることが示された。このノイズ抑制の性能はアーキテクチャのサイズに依存し、アーキテクチャが大きければノイズ抑制能力は高く、アーキテクチャサイズが小さければノイズ抑制能力も低くなる。

4 章の結果から、MACMSA は実データを学習することができることが示された。しかし、最適化済みシステムに学習データを入力した場合においても、アソシエイタにおける想起率は 100%でなかった。ただし、想起率が 100%でなかったとしても、学習データを再度システムに入力した際に得られる制御信号を実アームロボット上で実行した場合、制御不可能に陥ることはなく、一部のアームが伸びきっていないことが、目視で比べればわかる程度の違いに収まる。

5.2 考察

現在ロボット制御システムにおいて主流は深層学習を用いたしシステムである。深層学習はニューラルネットワークにおける近似能力と汎化性能を利用した枠組みである。深層学習における大きな功績は、大量の学習データを用いることで、観測点を増やし、認識力を飛躍的に向上させたことである。しかし、ニューラルネットワークの枠組みを使用している以上、観測点以外においてはどのような出力が得られるかは保証されていない。これはロボット制御に応用された場合に顕著である。実環境で動作するロボットは、センサー類にアナログ回路が含まれていることがほとんどであり、これに起因し全く同一状況であったとしても同一の入力を得ることは難しい。さらに当然のことながら、実環境では刻一刻と状況が変化するため同一状況が起こり得ると言うこと自体まれである。つまり、ロボット制御においては学習データである観測点と実際の入力が一致することはほとんどあり得ず、観測点外の動作が重要となる。

提案手法である MACMSA は強力なノイズ除去能力を有するホップフィールドネットワークを用いて、マルチモーダル処理が行われている。このホップフィールドネットワークにおけるノイズ処理能力は、単純に入力から出力を得る構造とはなっていない、独特なネットワーク構造に由来する能力である。ホップフィールドネットワークは入力されたパターンと記憶されたパターンを比較し、最も近いパターンを出力すると言う記憶ベースの動作である。また、エンコーダお

よびデコーダは認識力の高い深層学習の分野で発展したオートエンコーダを用いて学習されている。よって、MACMSA は深層学習を用いた高い認識力と、安定性の高い記憶ベースの両方を組み合わせた新しい制御アーキテクチャであると言えるだろう。

改良された MACMSA を用いた、アームロボットから得た実データの学習実験において、アソシエイタでの想起率が 100%でなかった。この事実は、すべての学習データを明記できるパターンへ変換することができなったことを示している。また、リアルタイム制御も行うことができなかった。4 章で議論した通り、これらの問題は、ハードウェアの選定または、エンコーダおよびデコーダとなるモジュールの変更が有効な手段であると考えられる。しかし、エンコーダの種類を変更する場合、情報ごとに異なるモデルの選択と実装は余儀なくされ、モデルのシンプルさは失わる。その結果、モデルの選択はアーキテクチャの設計者への負担となる。構造の変更を容易に行うのではなく、なぜオートエンコーダの一部として最適化されたエンコーダではリアルタイム制御ができなかったかを十分に検討する必要がある。その第一歩として、エンコーダからの出力がどのようなパターンになっているかの解析と、実データ自体の解析を行うことが重要である。実データを入力したエンコーダからの出力のハミング距離は解析した結果、極端に距離の短いパターンは検出されなかった。実データを解析したところ、実データは正規乱数による擬似データに比べ極端に分散が小さかった。しかし、分散を大きくする正規化方法を採用しても精度は向上しなかった。このことから、データの共分散も重要であると考えられる。したがって、分散共分散を調整することができるデータの白色化は有効な可能性が高い。

ロボット制御システムの中には深層学習を物体認識のみに使うシステムのように、システム全体ではなく、システムの一部として深層学習を含むシステムがある。このようなシステムでは、深層学習部は最適化されるが、その他の部分については既存のソフトウェア使うなど最適化が十分されていない場合がある。そのようなシステムでは、ソフトウェアのバージョンアップや、ハードウェアの一部変更などが起こると途端に制御不可の状態に陥る。実環境において、未知の状態が発生したり、意図しないノイズが混入したりと、システム設計者の想像を超える事態が起こり得る。そのため、システム全体を通してのノイズ耐性と安定性は重要であり、ノイズ耐性と安定性において大切なことはシステム全体の最適化度だと考えられる。MACMSA では、各オートエンコーダの最適化が完了した時点で、アソシエイタに銘記するパターンは唯一で決まる。したがって、オートエンコーダの最適化を以て、システム全体の最適化完了が期待できる。

近年では最新のモデルを使い、アーキテクチャがどんどん複雑化している。一方、提案した手法を構成するモジュールは使い古された古い手法ばかりである。

しかし、使い古された手法であるがそれらを組み合わせてシステムとして構築すると、高いノイズ耐性を示した。このことより、システムは最新で性能の良いモジュールで構築されることが最善とは限らず、モジュールの組み合わせが重要であると言える。

このシステム全体としての最適性は設計時点で方針を考慮する必要がある。提案された手法では、まずホップフィールドネットワークを中心とすることを選択した。これは提案手法がノイズ耐性を高めることを第一に考え、構造として、ノイズ耐性を持つネットワークを選択する必要があったためである。その後、ホップフィールドネットワークでマルチモーダルを扱うためにデメリットを克服するために必要なモジュールの選択を行った。このデメリットとは、ホップフィールドネットワークは 2 値を扱うモデルであるため、直接センサーからの情報を処理することができないことである。そのため、変換器として、シンプルなネットワークとして 3 層のフィードフォワードニューラルネットワークを採用した。しかし、フィードフォワードニューラルネットワークを学習するためには学習データが必要であるというさらなるデメリットが生まれる。この問題に対しては、オートエンコーダを用いてエンコーダ、デコーダを同時に学習することで解決を図った。最終的にモジュールを組み合わせた結果、アーキテクチャ全体を通してデメリットが解消されており、かつコンセプト通り稼働するか実験を通して確認を行った。このように、単に最新のモジュールを組み合わせるのではなく、目的に応じた構造を中心とし、その後、問題へ対応させる際に起こるデメリットのネストを深くし、すべてのデメリットが克服するようにアーキテクチャを設計するという方針が良いのではないかと考えられる。

本研究では計算コストを考え、オフライン学習を採用した。本来稼働し続けることが前提であるロボットにおいてオンライン学習は必須のように思える。しかし、人間にとって睡眠は学習を行うのに重要であるのと同様に、ロボットの学習においても睡眠のように学習に専念する時間を確保し、その間にロボットの学習を完了させるという手段もあっても良いのではないかと考えている。

これまでの既存手法は、実験で指定されたタスクに対して高い精度を得るという観点においては、一定の成功を収めているが、単に比較対象よりも高い精度を得ることを目的としているように思える。ノイズを含んだ入力を得たときに制御不能に陥ったシステムは危険なため、人間と相互作用する場面での利用は不可能である[80]。人間とのコミュニケーションにおいては、ノイズ除去能力を有する構造をアーキテクチャに取り入れ、実環境のノイズをシステムの構造を活用し如何に低減し、未知の状況に対応するかがより重要であると考えられる。

5.3 展望

実データにおけるエンコーダでの問題が解決すれば、リアルタイムでロボットの制御が可能になる。この段階で、実環境において、MACMSA を使用することができる。また、入力となる情報の種類を増やすことで、搭載されたセンサーの多いロボットを環境に合わせて、制御することができるであろう。さらに、人語などの情報も同時に処理することで、ロボットの動き制御だけではなく、自然言語処理を含めた相互作用を行える可能性がある。高い認識力と安定した動作を持つ MACMSA をベースとすることで、真に人間と相互作用ができるロボットの開発へ一歩近づくことができるだろう。

参考文献

1. 神崎 洋治, ロボット解体新書. SB クリエイト, 2017
2. 山田 誠二, 人とロボットの＜間＞をデザインする．東京電気大学出版局, 2007
3. Sakagami Y, Watanabe R, Aoyam C, Matsunaga S, Higaki N and Fujimura K, The intelligent ASIMO: System overview and integration. In *International Conference on Intelligent Robots and Systems*, 2002, 2478–2483.
4. Tanaka F, Isshiki K, Takahashi F, Uekusa M, Sei R and Hayashi K, Pepper learns together with children: Development of an educational application. In *15th International Conference on Humanoid Robots*, 2015, 270–275.
5. https://lovot.life/ (アクセス日 2021 年 12 月 9 日)
6. Bekey GA, *Autonomous Robots: From Biological Inspiration to Implementation and Control.* MIT Press, 2005.
7. 柿蔵 正義, 知能ロボット入門, 工業調査会, 1987
8. 太田 順, 倉林 大輔, 新井 民夫, 知能ロボット入門, コロナ社, 2001
9. 横小 路泰義, 稲邑 哲也, ロボティクス 第 8 章 行動を決定する．日本機械学会, 2011
10. Sünderhauf N, Brock O, Scheirer W, Hadsell R, Fox D, Leitner J, Upcroft B, Abbeel P, Burgard W, Milford M and Corke P, The limits and potentials of deep learning for robotics. *The International Journal of Robotics Research*, 2018, 37(4–5): 405–420.
11. Pierson HA and Gashler MS, Deep learning in robotics: A review of recent research. *Advanced Robotics.* 2017, 31(16), 821–835.
12. Suzuki K, Mori H and Ogata T, Motion switching with sensory and instruction signals by designing dynamical systems using deep neural network. *IEEE Robotics and Automation Letters,* 2018, 3(4), 3481–3488.
13. Saito N, Kim K, Murata S, Ogata T and Sugano S, Tool-use model considering tool selection by a robot using deep learning. *IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*, 2018, 270–276.
14. Sergeant J, Sünderhauf N, Milford M and Upcroft B, Multimodal deep autoencoders for control of a mobile robot. *Australasian Conference on Robotics and Automation,* 2015, 1–10.
15. Ngiam J, Khosla A, Kim M, Nam J, Lee H and Ng AY, Multimodal deep learning. In *Proceedings of the 28th International Conference on Machine Learning*, 2011, 689–696.
16. Chollet F, Xception: Deep learning with depthwise separable convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, 1251–258.

17. Qi CR, Su H, Mo K and Guibas LJ, PointNet: Deep learning on point sets for 3D classification and segmentation. *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, 652–660.
18. Chen Z, Jacobson A, Sünderhauf N, Upcroft B, Liu L, Shen C, Reid I and Milford M, Deep learning features at scale for visual place recognition. *2017 IEEE International Conference on Robotics and Automation,* 2017, 3223–3230.
19. Jain A, Tompson J, LeCun Y and Bregler C, MoDeep: A Deep Learning Framework Using Motion Features for Human Pose Estimation. Computer Vision –ACCV 2014, 2014, 302-315
20. Poria S, Cambria E, and Gelbukh A, Deep convolutional neural network textual features and multiple kernel learning for utterance-level multimodal sentiment analysis. *Conf. Empirical Methods Natural Language Process.*, 2015, 2539–2544.
21. Mukherjee S, Robertson N, Deep Head Pose: Gaze-Direction Estimation in Multimodal Video. *IEEE Transactions on Multimedia*, 2015, 17(11), 2094-2107
22. Liang M, Li Z, Chen T and Zeng J, Integrative Data Analysis of Multi-Platform Cancer Data with a Multimodal Deep Learning Approach. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2015, 12(4), 928-937
23. Radu V, Lane N, Bhattacharya S, Mascolo C, Marina M and Kawsar F, Towards multimodal deep learning for activity recognition on mobile devices. *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*, 2016, 185-188
24. Ding C, Tao D, Robust Face Recognition via Multimodal Deep Face Representation. *IEEE Transactions on Multimedia*, 2015, 17(11),2049-2058
25. Wu D, Pigou L, Kindermans P, Le N, Shao L, Dambre J and Odobez J, Deep Dynamic Neural Networks for Multimodal Gesture Segmentation and Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2016, 38(8), 1583-1597
26. Eitel A, Springenberg J, Spinello L, Riedmiller M and Burgard W, Multimodal deep learning for robust RGB-D object recognition. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015, 681-687
27. Kim T, Kang B, Rho M, Sezer S, Im E, A Multimodal Deep Learning Method for Android Malware Detection Using Various Features. *IEEE Transactions on Information Forensics and Security*, 2019, 14(3), 773-788
28. Liu W, Zheng W, Lu B, Emotion Recognition Using Multimodal Deep Learning. *ICONIP 2016: Neural information Processing*, 2016, 521-529
29. Leiva F, Lobos-Tsunekawa K, Ruiz-del-Solar J, Collision Avoidance for Indoor Service Robots Through Multimodal Deep Reinforcement Learning. *RoboCup 2019: Robot World Cup XXIII*, 2019, 140-153

30. Stefanidi A, Topnikov A, Tupitsin G and Priorov A, Application of Convolutional Neural Networks for Multimodal Identification Task. *2020 26th Conference of Open Innovations Association*, 2020, 423-428
31. Qiu X, Feng Z, Yang X and Tian J, Multimodal Fusion of Speech and Gesture Recognition based on Deep Learning, Journal of Physics: Conference Series, 2020, 1457(1)
32. Unlu H, Patel N, Krishnamurthy P and Khorrami F, Sliding-Window Temporal Attention Based Deep Learning System for Robust Sensor Modality Fusion for UGV Navigation. *IEEE Robotics and Automation Letters*, 2019, 4(4), 4216-4223
33. Park D, Hoshi Y and Kemp C, A Multimodal Anomaly Detector for Robot-Assisted Feeding Using an LSTM-Based Variational Autoencoder. *IEEE Robotics and Automation Letters* 2018, 3(3), 1544-1551
34. Lee M, Zhu Y, Srinivasan K, Shah P, Savarese S, Fei-Fei L, Garg A, Bohg J, Making Sense of Vision and Touch: Self-Supervised Learning of Multimodal Representations for Contact-Rich Tasks. *2019 International Conference on Robotics and Automation,* 2019, 8943-8950
35. Hori C, Hori T, Lee T, Zhang Z, Harsham B, Hershey J, Marks T and Sumi K, Attention-Based Multimodal Fusion for Video Description. *2017 IEEE International Conference on Computer Vision*, 2017, 4203-4212
36. Ma H, Li W, Zhang X Gao S, and Lu S, AttnSense: Multi-level Attention Mechanism For Multimodal Human Activity Recognition. IJCAI-19, 2019, 3109-3115
37. Levine S, Pastor P, Krizhevsky A, Ibarz J and Quillen D, Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International Journal of Robotics Research,* 2017, 37(4–5): 421–436.
38. Noda K, Arie H, Suga Y and Ogata T, Multimodal integration learning of robot behavior using deep neural networks. *Robotics and Autonomous Systems,* 2014, 62(6), 721–736.
39. Lenz I, Lee H and Saxena A, Deep learning for detecting robotic grasps. *The International Journal of Robotics Research,* 2015, 34(4–5), 705–724.
40. Yang P-C, Sasaki K, Suzuki K, Kase K, Sugano S and Ogata T, Repeatable folding task by humanoid robot worker using deep learning. *IEEE Robotics and Automation Letters,* 2017, 2(2), 397–403.
41. Gamal O Cai X and Roth H, Learning from Fuzzy System Demonstration: Autonomous Navigation of Mobile Robot in Static Indoor Environment using Multimodal Deep Learning *24th Int. Conf. Syst. Theory Control Comput.* 2020 , pp 218-225

42. Saito N, Ogata T, Funabashi S, Mori H and Sugano S, How to Select and Use Tools? : Active Perception of Target Objects Using Multimodal Deep Learning. *IEEE Robot. Autom. Lett.* 2021, 6 p 2517-2524
43. Yang L, Yan W and Wu H, Comparison of deep learning-based methods in multimodal anomaly detection: A case study in human–robot collaboration Science Progress. 2021, 2 p 1-24
44. 村山 龍太郎, 谷沢 智史, 西村一彦, Pepper プログラミング. SB クリエイティブ, 2015
45. Akhtar N and Milan A, Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access* 6, 2018, 14410–14430.
46. Ramachandram D and Taylor G, Deep Multimodal Learning, *IEEE Signal Processing Magazine*, 2017, 34(6),96-108.
47. Hinton G E, SALAKHUTDINOV R R, Reducing the Dimensionality of Data with Neural Networks. Science, 2006, 312(5786), pp 504-507
48. 熊沢 逸夫, 学習とニューラルネットワーク, 森北出版株式会社, 1998
49. Goodfellow I, Bengio Y and Courville A, DEEP LEARNING. The MIT Press, 2016
50. 吉富 康成, ニューラルネットワーク. 朝倉書店, 2002
51. Billard A, Dautenhahn K, and Hayes G, Experiments on human-robot communication with Robota, an imitative learning communicating doll robot. *Socially situated intelligence workshop*, 1998, 4-16
52. Jockel S, Mendes M, and Zhang J, Coimbra A.P, Crisostomo M, Robot navigation and manipulation based on a predictive associative memory. *IEEE 8th international Conference on Development and Leaning*, 2009, 1-7
53. Sudo A, Zheng C, Tsuboyama M, Sato A, and Hasegawa O, Pattern Based Reasoning Using Self-Organizing Incremental Neural Network. *IEICE TRANSACTIONS on Information and Systems*, 2008, 91(6), 1634-1647 [in Japanese]
54. Hu S, Liu Y, Liu Z, Chen T, Wang J, Yu Q, Deng L, Yin Y and Hosaka S, Associative memory realized by a reconfigurable memristive Hopfield neural network. *Nat. Commun*, 2015, 6, 7522
55. Huang P, Hasegawa O, Associative-memory-recall-based control system for learning hovering manoeuvres. *2015 International Joint Conference on Neural Networks*, 2015, 1-8
56. 伊藤 宏司, ニューロダイナミクス. 共立出版, 2010
57. Okada M (1996) Notions of associative memory and sparse coding. *Neural Networks,* 1996, 9(8), 1429–1458.
58. Finn C, Tan Y.X, Duan Y, Darrell t, Levine S, Abbeel P, Deep Spatial Autoencoder for Visuomotor Learning. *IEEE International Conference on Robotics and Automation,* 2016, 512-519

59. Yunchen P, Zhe G, Ricardo H, Xin Y, Chunyuan L, Andrew Stevens and Lawrence Carin, Variational Autoencoder for Deep Learning of Images, Labels and Captions. Advances in Neural Information Processing System, 2016, 29, pp 2352-2360
60. Zhuotun Z, Xinggang W, Song B, Cong Y, Xiang B, Deep Learning Representation using Autoencoder for 3D Shape Retrieval. Neurocomputing, 2016, 204, pp 41-50
61. Jaques N, Taylor S, Sano A and Picard A, Multimodal autoencoder: A deep learning approach to filling in missing sensor data and enabling better mood prediction. *International Conference on Affective Computing and Intelligent Interaction,* 2017, pp. 202-208
62. Lu X, Matsuda S, Hori C, Kashioka H, peech Restoration Based on Deep Learning Autoencoder with Layer-Wised Pretraining. *INTERSPEECH*, 2012, pp 1504-1507
63. Nurmaini S, Darmawahyuni A, Sakti Mukti AN, Rachmatullah MN, Firdaus F, Tutuko B, Deep Learning-Based Stacked Denoising and Autoencoder for ECG Heartbeat Classification. *Electronics,* 2020, 9(1), 135
64. Chen L, Zhou M, Su W, Wu M, She J, Hirota K, Softmax regression based deep sparse autoencoder network for facial emotion recognition in human-robot interaction. Information Sciences, 2018, 428, pp 49-61
65. 麻生 英樹, 多層ニューラルネットワークによる深層表現の学習, 人工知能学会誌, 2013, 28(4),649-659
66. Ng A, Sparse Autoencoder, CS294A Lecture notes https://web.stanford.edu/class/cs294a/sparseAutoencoder.pdf (アクセス日 2021 年 12 月 10 日)
67. 岡谷 貴之, 深層学習. 講談社 2015.
68. Srivastava N, Salakhutdinov R, Multimodal Learning with Deep Boltzmann Machines. Advances in Neural Information Processing System, 2012, 25, 2222-2230
69. Graves A Jaitly N, Towards End-To-End Speech Recognition with Recurrent Neural Networks. *PMLR,* 2014, 32(2), pp 1764 -1772
70. https://pytorch.org/ （アクセス日 2021 年 12 月 10 日）
71. 宮本 圭一郎,大川 洋平, 毛利 拓也, PyTorch ニューラルネットワーク 実装ハンドブック, 秀和システム,2018
72. Kingma D and Ba j 2014 Adam: A method for stochastic optimization arXiv (*Preprint* abs/1412.6980)
73. https://direct.sanwa.co.jp/ItemPage/400-CAM083 (アクセス日 2021 年 9 月 20 日)
74. https://kondo-robot.com/product/03157 (アクセス日 2021 年 9 月 20 日)

75. Zhang K, Zuo W, Chen Y, Meng D and Zhang Y, Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising. *IEEE Trans. Image Process.* 2017, 26, p 3142-3155
76. Ren S, He K, Girshick R, Sun J, Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 2015, 28, pp 91-99.
77. Albawi S, Mohammed T A and Al-Zawi S, Understanding of a convolutional neural network. *International Conference on Engineering and Technology*, 2017, pp. 1-6
78. Greff K, Srivastava R K, Koutník J, Steunebrink B R and Schmidhuber J, LSTM: A Search Space Odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, 2017 28(10), pp 2222-2232
79. Li J, Mohamed A, Zweig G and Gong Y, LSTM time and frequency recurrence for automatic speech recognition. *IEEE Workshop on Automatic Speech Recognition and Understanding*, 2015, pp 187-191
80. 岡村 親宜，ロボットは人間に危害を加えてはならない．労働基準調査会，1985

研究業績

学位に関する業績

学術論文

[1]**Motohiro Akikawa**, Masayuki Yamamura, Materializing Architecture for Processing Multimodal Signals for a Humanoid Robot Control, 2021, JACIII, Vol25(3), pp.335-345 (ITCA2020 推薦論文として投稿)
(第 2 章、第 3 章に対応)

会議

[2]**Motohiro Akikawa**, Masayuki Yamamura, Learning reaching tasks using an arm robot equipped with MACMSA, 2021, ISAIC
(第 4 章に対応)

その他の業績

学術論文

[3]Kazuki Takabatake, Kazuki Izawa, **Motohiro Akikawa**, Keisuke Yanagisawa, Masahito Ohue, Yutaka Akiyama: Improved large-scale homology search by two-step seed search using multiple reduced amino acid alphabets, 2021, Genes, 12: 1455

会議(査読あり)

[4]Kazuki Takabatake, Kazuki Izawa, **Motohiro Akikawa**, Keisuke Yanagisawa, Masahito Ohue, Yutaka Akiyama, Improved Homology Search for Metagenomic Analysis by Two-Step　Seed Search with Reduced Amino Acid Alphabet, 2021, ICBBS2021(Accepted as an oral presentation)
[5]**Motohiro Akikawa**, Masayuki Yamamura, Feasibility Architecture for Processing Multimodal Signals for a Humanoid Robot Control, 2020, ITCA (BestPaper Award 受賞)

会議(査読なし)

[6]高畠和輝, 伊澤和輝, **秋川元宏**, 大上雅史, 秋山泰: 圧縮アミノ酸を利用した二段階のシード探索によるメタゲノム配列相同性検索の改良, 情報処理学会研究報告, 2020, 2020-BIO-61(10), pp.1-6
[7]**秋川元宏**, 山村雅幸, マルチモーダル連想記憶を用いたお友達ロボットの設計, 2013, 第 40 回知能シンポジウム

総説・解説
[8]現代化学「コンピュータは魔法の箱？」連載　全 6 回

1.安田翔也, **秋川元宏**, 山村雅幸, 私にもシミュレーションって関係あるの 2017 年 1 月, pp.50-52
2.安田翔也, **秋川元宏**, 山村雅幸, サンプルプログラムを実行してみよう, 2017 年 2 月, pp.44-48
3.**秋川元宏**, 安田翔也, 山村雅幸, 拡散現象をシミュレーションしてみよう , 2017 年 3 月, pp.62-64
4.安田翔也, **秋川元宏**, 山村雅幸, 微分方程式を使ったシミュレーション, 2017 年 4 月 pp.46-49
5.**秋川元宏**, 山村雅幸, 分子の動きを観察してみよう, 2017 年 5 月, pp.37-40
6.**秋川元宏**, 安田翔也, 山村雅幸, 計算速度と使用ツール, 2017 年 6 月, pp.55-58

謝辞

長きに渡り辛抱強くご指導いただきました山村雅幸教授には心よりお礼申し上げます。研究を開始した当初は国内での深層学習ブームはまだ起こっておらず、これからニューラルネットワークの研究を行うのは厳しいと言う雰囲気が分野にはあったのにもかかわらず、ニューラルネットワークの研究をしたいと言う私の意思を尊重してくださるばかりか、適切なご指導、支援をいただき誠にありがとうございました。真に行いたい研究を遂行することができました。また、山村教授の柔軟な思考、幅広い興味に刺激を受け、研究者はどうあるべきなのかを理解することができました。頂いた経験、知識を今後もフル活用して、今後も精進してまいります。

学生時代においてはリサーチアシスタントとして、その後は研究員として、秋山泰教授と関わりを持たせていただきました。秋山教授には勤務のこと以外にも研究や進路などさまざまな相談に乗っていただき、精神的に何度も助けていただきました。心よりお礼申し上げます。

大上雅史助教には、研究の合間に研究活動に関するご相談をさせていただき、学会運営とはどう言うものなのかなど、今度も研究者として活動するために非常に役立つ知識をいただきました。心よりお礼申し上げます。

柳澤渓甫助教とは良き友人として、研究で心が折れた時に幾度となく相談に乗っていただきました。心よりお礼申し上げます。

山村研究室に所属していた先輩、後輩の皆様にも刺激をもらい、時にははげまされ、これまで研究を続けることができました。お礼を申し上げます。

最後に両親に向けて。父からは、自分の好きなように生きなさいと常に言われ、その言葉が、私が研究者になりたいと思った時、なかなか研究がうまくいかず諦めようとした時、自分はどのように道を進めたらいいのかに悩んだ時、自分はどのように生きたいのかを再考するきっかけになりました。また、母は体調だけは気をつけてと、常に気遣ってくれました。両親に心より感謝します。