令和４年度

修　士　論　文

「一次遅れ＋むだ時間系」システムの変動検出および分類のためのLSTMを用いた多段階モデルの開発

電気システム制御プログラム　　社会情報学研究室
M212351　　森　仁紀

指導教員

教　授　　西崎　一郎
准教授　　林田　智弘
助　教　　関崎　真也

令和５年２月２１日

広島大学大学院先進理工系科学研究科

あらまし

PID 制御はプロセス制御の現場において主力の制御方法であり，適切な制御パラメータを設定することで，目標値に追従する出力を得ることができる．一方で，経年劣化といった外的要因や内部構造の特性の変化などによりシステムパラメータが変動した場合には，制御パラメータを再設定する必要があるが，システムの特性が変動している対象に対して，適切に制御パラメータを設定することは困難である．そのため，システム変動の種類を推定するとともに，どのパラメータがどの程度変動したのかについて検知することで，制御パラメータを再設定することが必要になる．本論文では，機械学習の一種であり，内部に信号をフィードバックする構造を持つリカレントニューラルネットワークのなかでも，長期の時系列特性を持つデータの取り扱いに適した LSTM(Long-Short Term Memory) を用いて，複数のパラメータが同時に変動するシステムに対してシステム変動の種類を分類するための手法を提案し，数値実験によって提案手法の有効性を示す．

目 次

第 1 章

はじめに

制御対象へ適切な入力信号を与えることで，所望のシステム出力を得ることは制御分野の目的の一つであり，エアコンの温度制御，電力系統の電圧制御，化学プラントや製造工程の温度制御や流量制御まで幅広い分野で制御の技術が応用されている．特に，プラント業界においては，制御対象からの出力を目標値に到達させ，追従させることは，製造物の質の向上や製造にかかるコストの省エネルギー化，不良品の削減につながるなど，非常に重要な制御目的の一つである．制御対象に対応した制御パラメータを適切に設定することで，所望の制御性能を発揮することができる．一方で，外的要因によって制御対象のシステムパラメータが変化した場合，事前に設計していた制御パラメータでは制御対象からの出力値が目標値を逸脱してしまい，本来のパフォーマンスが低下してしまう場合が存在する．このとき，セルフチューニング制御 [1,2] のようにオンラインで制御パラメータの調整をこまめに行うことも有効な手段ではあるが，システム変動が起きていない場合にもパラメータの調整を行うため，パラメータ調整コストが多くかかる問題が存在している．そのため，制御対象のシステムパラメータが変動した場合のみ，制御パラメータの再調整を行うことでパラメータ調整のコスト削減を行う必要がある．

山本 [3] により，制御性能を改善するための手法の一つとして,「制御性能評価機構」と「制御パラメータ調整機構」の二つの機構を用いた手法を提案している.「制御性能評価機構」では，逐次最小二乗法を用いた制御対象もとに設計を行った設計モデルの制御性能の評価を行っており，入出力データから予測される未来の応答をもとに設計したモデルを構築し，そのモデルによる予測精度が高ければ適切な制御結果を得ることができると判断する．反対に，予測精度が悪ければ何らかの要因により制御対象の内部特性が変化し制御性能が悪化したものと評価する．また，林田ら [4] は，生物の脳の働きの一部を数式でモデル化したニューラルネットワーク (NN: Neural Network) の一種であり，情報を逆伝播する構造をもち，音声データの識別や，画像データの認識などの時系列性が存在するデータの特徴の学習・予測に適したリカレントニューラルネットワーク（RNN: Recurrent NN）を用いることで，線形の入出力関係を持つ比較的単純なシステムに対するシステム変動検出手法を提案しており，シミュレーションによる数値実験により，高精度にシステムの変動を検知することに成功している．水口ら [5] は RNN の一種であり，時系列的特徴を持つデータの学習において高い汎化能力を持つ exMCRNN(extended Multi-Context Recurrent Neural Network)[7] を用いることで，単一のパラメータのシステム変動での比較的

単純なシステム変動分類に成功している．変動検出をした際に，どのシステムパラメータがどの程度変動したのかをある一定のレベルで特定することができれば，その後，システム変動をもとに制御パラメータを再度適切に調整・設計することが容易になり，所望の制御性能の発揮が見込める．

NN は，生物の神経系の機能に着目し，工学的に数式モデルへと応用したものである．NN の研究は 1940 年代まで遡ることができ，文字認識 [9]，音声認識 [10] のようなパターン認識の分野や株価予測 [11]，気象予測 [12] のような時系列データ予測の分野など幅広い分野で応用されている．

本論文では，RNN を応用することによりシステム変動の種類を分類可能である点 [4] と，exMCRNN では長期の時系列特性を持つデータの学習をした際に古い情報が消失してしまい適切にデータの時系列性を学習することが困難な場合がある点を考慮し，長期的な時系列性のあるような時系列データに対して，exMCRNN よりも高い学習性能を持つ LSTM(Long Short Term Memory)[6] を用いたシステム変動の分類手法 [8] を応用して，複数のパラメータが同時に変動する場合の分類手法を提案する．提案手法では，シミュレータによって，システムの特性を決定づけるいくつかのシステムパラメータを同時に変動させたデータを用いて，LSTM にシステム変動の特徴を学習させ，LSTM からの出力値を用いて分類することを行う．また，学習データとして複数ステップの入出力データを一度に与えることによってステップ間の特徴を学習させる．さらに，一つの LSTM を用いて分類することが困難な入出力データに関しては，適切に分類することができない異なる特性を持つデータを似た特徴を持つデータであると解釈し，これらを一つのグループとして，同じグループに属するデータのみを用いて，他の LSTM の学習を行うような多段階のモデルを用いて分類を行う．このような方法で複数のパラメータが同時に変動するシステムに対してシステムの変動を検出し分類することで提案手法の有効性を示す．

本論文の構成を以下に示す．第 2 章では，本論文で考える制御対象と本論文で用いる制御法である PID 制御について説明する．第 3 章では，ニューラルネットワークについての説明と，それを用いたシステム変動の分類手法について説明する．第 4 章では，シミュレータを用いた数値実験によるシステム変動の分類結果を示す．第 5 章では，本論文のまとめと今後の課題について述べる．

第 2 章

制御対象と PID 制御

本章では，本論文で対象とする制御対象および PID 制御について説明する．

2.1. 対象モデル

本論文では，制御対象を「一次遅れ＋むだ時間」系のシステムとし，制御対象の数式モデルは次式で表される．

$$G(s) = \frac{K}{1+Ts}e^{-Ls} \tag{2.1}$$

ここで，K はシステムゲイン，T は時定数，L はむだ時間，s は複素数を表す．各システムパラメータは制御対象の定常値，応答の速さ，入力値が出力値に反映されるまでの時間の遅れに関係するパラメータである．また，式 (2.1) に対応したサンプリング間隔 T_s で離散化した CARIMA（Controlled Auto-Regressive and Integrated Moving Average）モデルを式 (2.2) に示す．

$$A(z^{-1})y(t) = z^{-(d+1)}B(z^{-1})u(t) + \xi(t)/\Delta \tag{2.2}$$

ただし，$A(z^{-1})$，$B(z^{-1})$ は次式で与えられる．

$$\left.\begin{aligned} A(z^{-1}) &= 1 + a_1 z^{-1} + \cdots + \cdots + a_n z^{-n} \\ B(z^{-1}) &= b_0 + b_1 z^{-1} + \cdots + b_m z^{-m} \end{aligned}\right\} \tag{2.3}$$

$u(t)$ は制御対象への入力，$y(t)$ は制御対象からの出力，d は離散時間系におけるむだ時間，$\xi(t)$ は平均 0，分散 σ_ξ^2 で表されるガウス性白色雑音である．また，時間遅れ演算子を z^{-1} で表し，$z^{-1}y(t) = y(t-1)$ で表現される．本論文では一次遅れ系を対象とするため，$n = m = 1$ である．n, m はそれぞれ $A(z^{-1}), B(z^{-1})$ の次数を表現している．Δ は差分演算子を表しており，$\Delta = 1 - z^{-1}$ で定義される．また，コントローラ $C(z^{-1})/\Delta$ を次節で述べる PID 制御器として以下の式で与える．

$$u(t) = \frac{C(z^{-1})}{\Delta}\{r(t) - y(t)\} \tag{2.4}$$

$$C(z^{-1}) = K_P\Delta + K_I + K_D\Delta^2 \tag{2.5}$$

ただし，$r(t)$ は目標値を表しており，制御パラメータ K_P，K_I，K_D はそれぞれ，比例ゲイン，積分ゲイン，微分ゲインである．

2.2. PID 制御

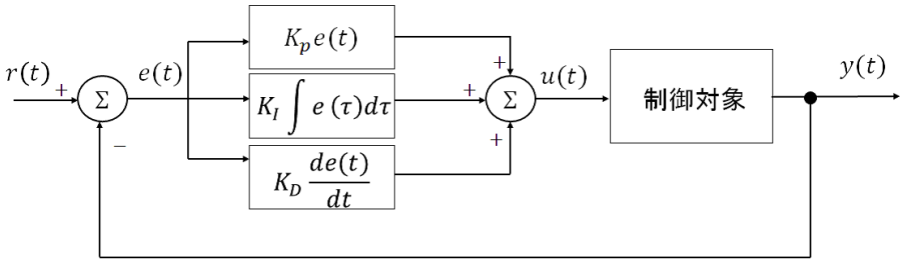

図 2.1: PID 制御のブロック線図

PID 制御は，フィードバック制御の一種であり，一般に図 2.1 のようなブロック線図で表現される．フィードバック制御の基本的な考え方は出力値 $y(t)$ が目標値 $r(t)$ に到達していなければ，入力値 $u(t)$ を大きくし，逆に出力値が目標値より大きければ，入力値を小さくすることである．PID 制御は，出力値 $y(t)$ と目標値 $r(t)$ との偏差 $e(t)$ により入力値 $u(t)$ を決定し，制御する比例制御 (P 制御)，偏差の積分により制御する積分制御 (I 制御)，偏差の微分により制御する微分制御 (D 制御) の 3 つの要素によって制御を行う．PID 制御は構造が単純なこともあり，プロセス制御の現場では今なお主力の制御法である [13]．以下では，PID 制御の各要素について詳しく述べる．

2.2.1. 比例制御 (P 制御)

P 制御は，現時刻の制御対象からの出力値 $y(t)$ と目標値 $r(t)$ から偏差 $e(t)$ を求め，偏差をフィードバックし，それに比例した入力値 $u(t)$ を制御対象への入力として，制御を行う．このとき，入力値 $u(t)$ は式 (2.6) で与えられる．

$$u(t) = K_P e(t) \tag{2.6}$$

多くの場合 K_P を大きくすると，応答が振動的になり，さらに大きくすると発散し，不安定な系になる．また，偏差が小さくなると，偏差と比例の関係にある入力値が消失し，偏差が 0 にならない状況がある．このときの偏差を定常偏差 (図 2.2) といい，そのため，P 制御だけでは完全に偏差をなくすことはできない場合がある．

2.2.2. 積分制御 (I 制御)

I 制御は，過去から現在までの偏差の積分値をフィードバックする制御である．小さな偏差であっても積分することで大きな値になるため，偏差が 0 に近くとも入力値が消失しない特徴を持っている．P 制御と I 制御を組み合わせた PI 制御によって，

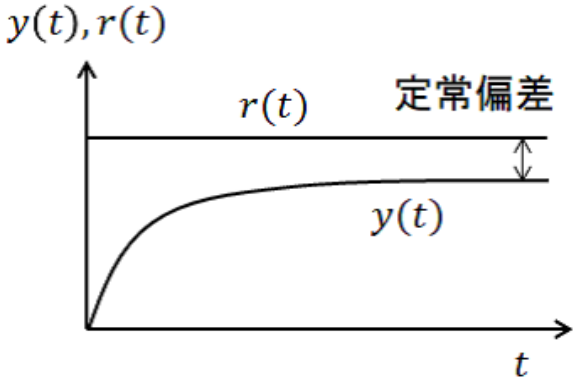

図 2.2: P 制御

定常偏差を解消することができる．このとき，入力値 $u(t)$ は以下の式で与えられる．

$$u(t) = K_P e(t) + K_I \int_0^t e(\tau) d\tau \tag{2.7}$$

ただし，I 制御は過去の制御対象の状態に基づいて制御を行うため，出力値の急激な変化に対する応答の遅れが生じてしまうことがあり，目標値から外れやすくなる場合がある (図 2.3)．

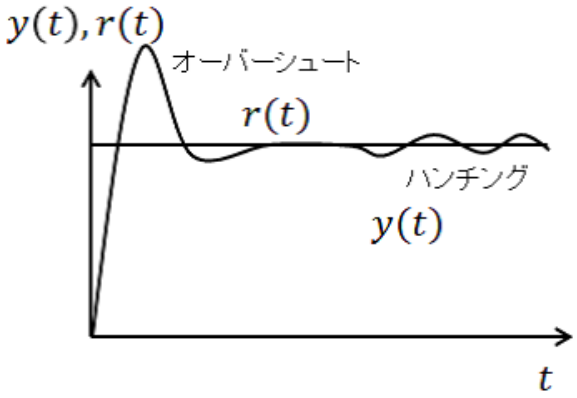

図 2.3: PI 制御

2.2.3. 微分制御 (D 制御)

D 制御は，偏差の微分値をフィードバックする制御であり，未来の状態を予測した制御を行うことができる．PI 制御と組み合わせた PID 制御によって，早めに偏差の増減を抑制し，PI 制御では困難であった出力値の急激な変化に対する応答の遅れ

を改善することができる．このとき，入力 $u(t)$ は以下の式で与えられる．

$$u(t) = K_P e(t) + K_I \int_0^t e(\tau)d\tau + K_D \frac{de(t)}{dt} \tag{2.8}$$

ただし，K_D を大きくしすぎると，ノイズの影響を受けやすくなり，早く振動するようになってしまう．

PID 制御では，K_P, K_I, K_D の PID パラメータを適切に決定することで，図 2.4 のように出力値が目標値に追従するようになり，所望の制御性能を得ることができる．

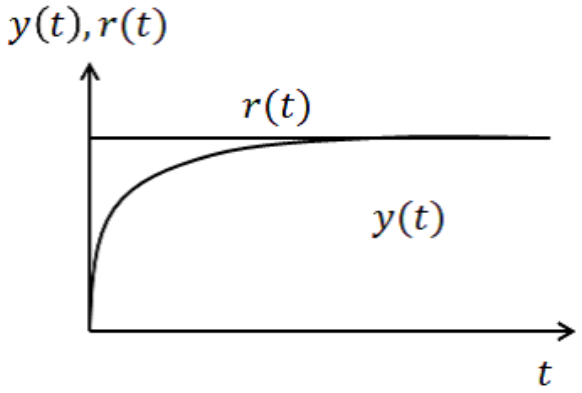

図 2.4: PID 制御

2.3. 制御パラメータの調整

制御パラメータを調整する手法として，試行錯誤しながら調整する手法や ZN(Ziegler and Nichols) 法 [14]，CHR(Chien, Hrons, Reswick) 法 [15] などの手法がある．本論文では，文献 [16] で提案されている閉ループデータから制御パラメータを算出する手法を用いることで制御パラメータの算出を行う．この手法は最小分散制御則に基づく制御性能評価 (Minimum Variance Index： MV-Index)[17] と Fictitious Reference Iterative Tuning（FRIT）[18] によって適切な制御パラメータを算出する手法である．以下でその詳細について述べる．

2.3.1. 最小分散制御ベンチマーク

式 (2.2) で表現されるシステムに対して，最小分散制御則を以下の評価規範の最小化に基づいて導出する [19].

$$J = E[\phi^2(t + d + 1)] \tag{2.9}$$

$\phi(t+d+1)$ は制御対象からの出力値と参照モデルからの出力値から求められる誤差を示しており，次式で与えられる．

$$\phi(t+d+1) := P(z^{-1})y(t+d+1) - P(1)r(t) \tag{2.10}$$

ここで，$P(z^{-1})$ は設計多項式であり，文献 [2] の手法に基づいて設計される．
次に，式 (2.11) の Diophantine 方程式を導入する．

$$P(z^{-1}) = \Delta A(z^{-1})E(z^{-1}) + z^{-(d+1)}F(z^{-1}) \tag{2.11}$$
$$E(z^{-1}) = 1 + e_1 z^{-1} + \cdots + e_d z^{-d} \tag{2.12}$$
$$F(z^{-1}) = f_0 + f_1 z^{-1} + \cdots + f_n z^{-n} \tag{2.13}$$

$\xi(t)$ を白色雑音，$A(z^{-1},)B(z^{-1})$ をシステムパラメータ，$C(z^{-1})$ をコントローラとすると，式（2.10）は次式のように書き換えられる．

$$\phi(t+d+1) = E(z^{-1})\xi(t+d+1) + S(z^{-1})\xi(t) \tag{2.14}$$
$$T(z^{-1}) := \Delta A(z^{-1}) + z^{-(d+1)}B(z^{-1})C(z^{-1}) \tag{2.15}$$
$$S(z^{-1}) := \frac{F(z^{-1}) - B(z^{-1})C(z^{-1})E(z^{-1})}{T(z^{-1})} \tag{2.16}$$

式 (2.15) を用いると評価規範 J は次式となる．

$$J = E\left[E(z^{-1})\xi(t+d+1)^2 + S(z^{-1})\xi(t)^2\right] \tag{2.17}$$

最小分散制御則は，式 (2.9) で表される誤差を最小化することである．したがって，コントローラ $C(z^{-1})/\Delta$ によって式 (2.15) の $S(z^{-1})$ を 0 にすることが必要になる．また，(2.17) 式は $\xi(t)$ が白色雑音であることから，次式のように分離することができる．

$$J = E\left[\{E(z^{-1})\xi(t+d+1)\}^2\right] + E\left[\{S(z^{-1})\xi(t+d+1)^2\}\right] \tag{2.18}$$
$$= J_{\min} + J_0 \tag{2.19}$$

となる．ここで，適切にパラメータの設計を行ったコントローラを用いた場合，誤差の値が最小化，つまり $J_0 = 0$ となり，最小分散となる．

2.3.2. 制御性能評価指標に基づいた制御パラメータの調整

最小分散制御則に基づく制御性能評価指標 (MV-Index)κ を次式で定義する．

$$\kappa := \frac{J_{\min}}{J_{\min} + J_0} \tag{2.20}$$

$J_0 = 0$ となる場合が最小分散制御則を達成することができ，$\kappa \to 1$ のとき適切なパラメータ設計がなされ最適な制御がなされ他と評価することができ，$\kappa \to 0$ のと

き最適な制御系の設計ができていないと評価できる．ここで，$J_{\min}$ を算出するには，多項式 $E(z^{-1})$ のパラメータを得る必要がある．そこで，閉ループデータから制御性能評価指標 κ を直接算出するために，文献 [20] による手法で得られたデータを AR モデルで近似的に表現し，次式について考える．

$$\phi(t) - \bar{\phi} = \epsilon(t) + \sum_{i=0}^{M} \alpha_i \{\phi(t-d-i) - \bar{\phi}\} \tag{2.21}$$

$$\epsilon(t) := E(z^{-1})\xi(t) \tag{2.22}$$

ここで，$\bar{\phi}$ は $\phi(t)$ の平均，α_i は自己回帰パラメータ，M はその次数である．パラメータ α_i は N 個のデータを用いて最小二乗法により算出する．

$$\tilde{\boldsymbol{\Phi}}(t) = \boldsymbol{X}(t)\boldsymbol{\alpha}(t) + \boldsymbol{\Xi}(t) \tag{2.23}$$

$$\tilde{\phi}(t) := \phi(t) - \bar{\phi} \tag{2.24}$$

$$\tilde{\boldsymbol{\Phi}}(t) := [\tilde{\phi}(t), \tilde{\phi}(t-1), \cdots, \tilde{\phi}(t-N+1)]^T \tag{2.25}$$

$$\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \cdots, \alpha_M]^T \tag{2.26}$$

$$\boldsymbol{\Xi}(t) = [\epsilon(t), \epsilon(t-1), \cdots, \epsilon(t-N+1)]^T \tag{2.27}$$

$$\boldsymbol{X}(t) = \begin{bmatrix} \phi(t-\tilde{d}-1) & \cdots & \tilde{\phi}(t-d-M) \\ \tilde{\phi}(t-d-2) & \cdots & \tilde{\phi}(t-d-M-1) \\ \vdots & \ddots & \vdots \\ \tilde{\phi}(t-d-M) & \cdots & \tilde{\phi}(t-d-M-N+1) \end{bmatrix} \tag{2.28}$$

したがって，パラメータ $\boldsymbol{\alpha(t)}$ は次式で算出される．

$$\boldsymbol{\alpha}(t) = \{\boldsymbol{X}(t)^T\boldsymbol{X}(t)\}^{-1}\boldsymbol{X}(t)^T\tilde{\boldsymbol{\Phi}}(t) \tag{2.29}$$

このとき，式 (2.20) はデータ数 N を十分大きくすることによって，(2.30) 式で与えられる．

$$\kappa = \frac{\{\tilde{\boldsymbol{\Phi}}(t) - \boldsymbol{X}(t)\boldsymbol{\alpha}(t)\}^T\{\tilde{\boldsymbol{\Phi}}(t) - \boldsymbol{X}(t)\boldsymbol{\alpha}(t)\}}{\tilde{\boldsymbol{\Phi}}(t)^T\tilde{\boldsymbol{\Phi}}(t)} \tag{2.30}$$

FRIT の計算のなかにこの制御性能評価指標 κ を取り入れる．FRIT は，一回の実験によって得られた入出力データ $u_0(t)$，$y_0(t)$ および，これらのデータから生成される疑似参照入力 $\tilde{r}(t)$ によって制御パラメータを導出する方法である．

図 2.5 に，FRIT のブロック線図を示す．このとき，得られた入出力データ $u_0(t)$，$y_0(t)$ と $C(z^{-1})$ の関係式は次式となる．

$$u_0(t) = \frac{C(z^{-1})}{\Delta}\{\tilde{r}(t) - y_0(t)\} \tag{2.31}$$

式 (2.31) より疑似参照入力 $\tilde{r}(t)$ は，制御器と実験データから以下のように算出できる．

$$\tilde{r}(t) = C(z^{-1})^{-1}\Delta u_0(t) + y_0(t) \tag{2.32}$$

このとき，出力データ $y_0(t)$ を用いて，$\phi_0(t)$ が算出できる．さらに，この $\phi_0(t)$ を用いて，一回の実験データにおける制御性能評価指標 κ_0 が得られる．同様に，参照モデルからも疑似参照制御性能評価指標 $\tilde{\kappa}_m$ を得る．FRIT により，κ_0 と $\tilde{\kappa}_m$ の絶対誤差を最小化するように，制御パラメータを決定する．

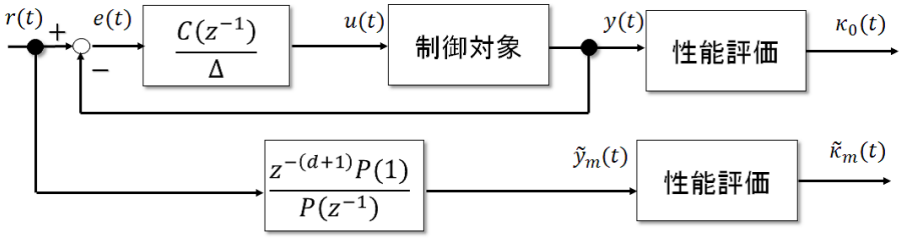

図 2.5: FRIT のブロック線図

第 3 章

ニューラルネットワークによるシステム変動の分類

ニューラルネットワークは人間を筆頭とする生物の情報処理システムの中枢である神経系の特徴的な機能に着目して，工学的にモデル化をしたものである．ニューラルネットワークの研究は 1940 年代にまで遡ることができ，様々な分野における数多くの応用成功例の報告がある．

3.1. 神経細胞と形式ニューロン

生物の神経系は，多数の神経細胞 (ニューロン) がシナプス (他のニューロンとの接合部) をもち，それぞれが並列処理を行うことによって情報の処理・伝達を行っている．ニューロンの構成要素として樹状突起，軸索の二つの要素が存在する．

各ニューロンの樹状突起は，シナプスを通して他のいくつかのニューロンからの入力信号を受け取る．その後，細胞体にて出力信号の合成を行い，軸索を通して他のニューロンへと情報が出力していく．以上の流れを通して複数のニューロン同士がネットワークを形成し複数の情報を同時に処理できるようになる．

以上の人間の脳内の情報処理原理を単純化しモデル化したのが，McCulloch and Pitts[21] により提案された，形式ニューロンであり，任意の論理関数が計算可能であることを示した．多入力 - 1 出力の形式ニューロンを，図 3.1 に示す．

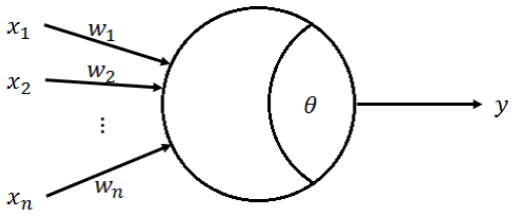

図 3.1: 形式ニューロン

図 3.1 において，$x_i, w_i (i = 1, 2, \ldots, n)$ はそれぞれ，入力変数，その入力に対する重みであり，y は出力，θ はニューロンの閾値である．このようなニューロンの出力 y は次式に基づく．

$$y = f\left(\sum_{i=1}^{n} w_i x_i - \theta\right) \tag{3.1}$$

ここで，$f(\cdot)$ は活性化関数と呼ばれ，基本的には次式のようなステップ関数が用いられる．

$$f(x) = \begin{cases} 1 & \text{if } x \geqq 0 \\ 0 & \text{if } x < 0 \end{cases} \tag{3.2}$$

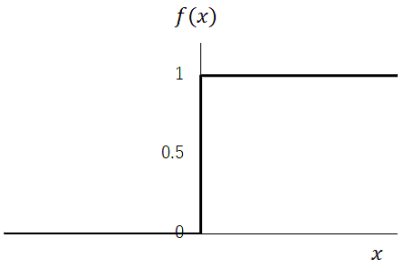

図 3.2: ステップ関数

すなわち，式 (3.1) は，重み付けられた入力の和が閾値を超えた場合は 1 を出力し，閾値未満の場合は 0 を出力することを表している．

近年では，活性化関数として式 (3.3) のようなシグモイド関数がしばしば使用される．これにより，出力値が多様化し，実数データなどに適したモデルとなる．

$$\sigma(x) = \frac{1}{1 + \exp(-x)} \tag{3.3}$$

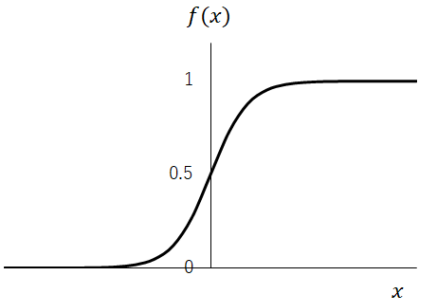

図 3.3: シグモイド関数

3.2. ニューラルネットワークのモデル

先行研究にて人口ニューロンを複数結合させたニューラルネットワークのモデルが多く開発されている．本節にていくつかのニューラルネットワークのモデルについて説明する．

3.2.1. Feed-forword Neural Network (FNN)

図 3.4 は Feed-forword Neural Network (FNN) と呼ばれるモデルであり，全てのユニットにおいて順方向にのみ信号が伝えられる．また，ユニットは層構造で形成されており，最初に入力が与えられるユニットで構成される層を入力層，最終的な信号を出力する層を出力層，それらの間で信号を伝達させていくユニットで構成される層を中間層，または隠れ層と呼ぶ．中間層は複数存在し得る．

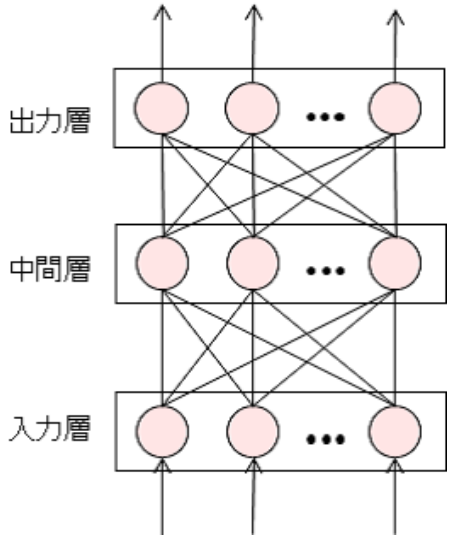

図 3.4: Feed-forword Neural Network

一方，ユニット間の信号は順方向のみではなく順方向と逆方向の双方向に伝播するものも存在し，双方向伝播型モデルであるニューラルネットワークも存在する．

これらのモデル化により，線形分離不可能である，入力変数と出力変数との関係を解析することができる．

3.2.2. Recurrent Neural Network (RNN)

多くのデータマイニング問題では，時系列的特徴を持つデータを取り扱う．入力データが時系列的特徴を持つデータである場合，過去のデータを含めて学習を行ったほうが良い結果が得られる．しかし，図 3.4 のようなニューラルネットワークモデルでは現在の入力データのみを考慮しているため，時系列データの取り扱いに適

しているとは言えない．そこで，時系列データを扱う場合，再帰的な構造を持つリカレントニューラルネットワーク (RNN： Recurrent Neural Network）と呼ばれるモデルが用いられる．RNN のモデルの代表的な例として，Jordan Network[22] と Elman Network[23] がある．それぞれ図 3.5，3.6 に示す．

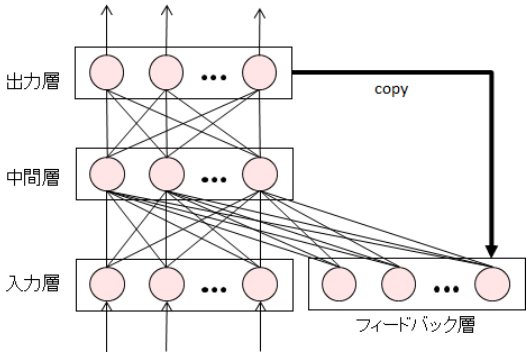

図 3.5: Jordan Network[22]

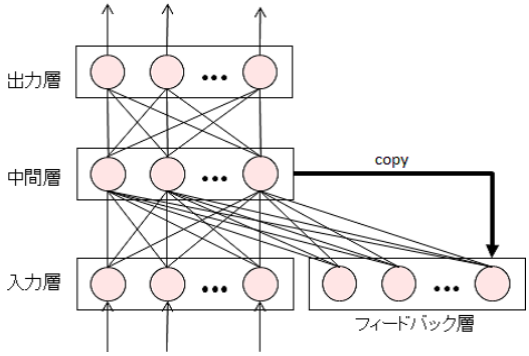

図 3.6: Elman Network[23]

Jordan Network は，図 3.5 のように，出力データを入力層へフィードバックさせるモデルである．つまり，フィードバック層には 1 期前の出力層からのデータが保持されている．

一方，Elman Network は，図 3.6 のように，中間層からの出力を入力層へフィードバックさせたモデルである．つまり，フィードバック層には 1 期前の中間層からの出力値が保持されている．また，Elman Network の拡張し，フィードバック層を複数個としたモデルを，Elman Tower Network という．

これらのネットワークは，図 3.4 のような階層構造をもつニューラルネットワー

クに加えフィードバック層を持っており，これにより，現在と過去の情報をもとに算出されたデータも出力に反映可能なため，様々な時系列データの予測や分類に適用されている．

3.2.3. Time Delay Neural Network (TDNN)

Time Delay Neural Network (TDNN)[24] は，RNN と同様，時系列データを扱うのに適したモデルであり，音声認識 [24] や株価予測 [25] に適用されている．

TDNN は，図 3.7 で表されるように，過去複数期分の入力層をコピーし，コピーした入力層からのデータを再度入力として利用し，出力に反映することができるモデルである．

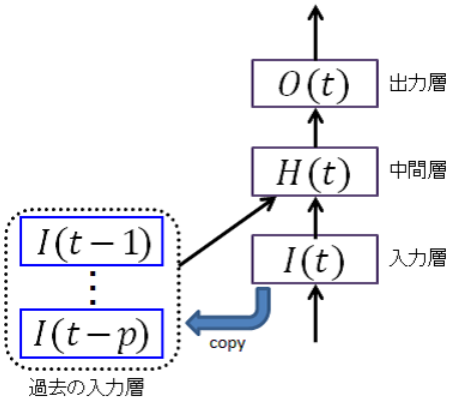

図 3.7: Time Delay Neural Network[24]

3.2.4. Multi-Context Recurrent Neural Network(MCRNN)

Multi-Context Recurrent Neural Network(MCRNN) は，Huang *et al.*[26] により提案された，RNN のモデルの一つである．図 3.8 のように，Elman Tower Network と同様に数期前までの中間層データが保持されたフィードバック層を複数個持っており，そのフィードバック層は中間層だけでなく，出力層へも結合している．文献 [26] において，電力負荷予測や手書き文字認識などといった時系列データで実験を行い，MCRNN が Elman Network や Elman Tower Network よりも優れていることが示された．

3.2.5. extended MCRNN (exMCRNN)

Katagiri *et al.*[27] は，入力層を複数期コピーすることによって過去の情報を含めた学習ができる TDNN の構造を MCRNN に組み込み，extended MCRNN(exMCRNN) とした．このモデルは，時系列データの解析において，MCRNN に比べ高い汎化能

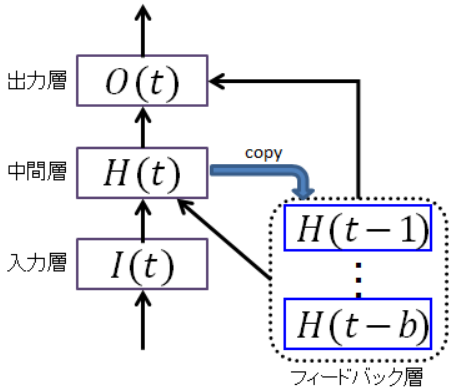

図 3.8: Multi-Context Recurrent Neural Network[26]

力を持つことが示された．exMCRNN の構造を図 3.9 に示す．

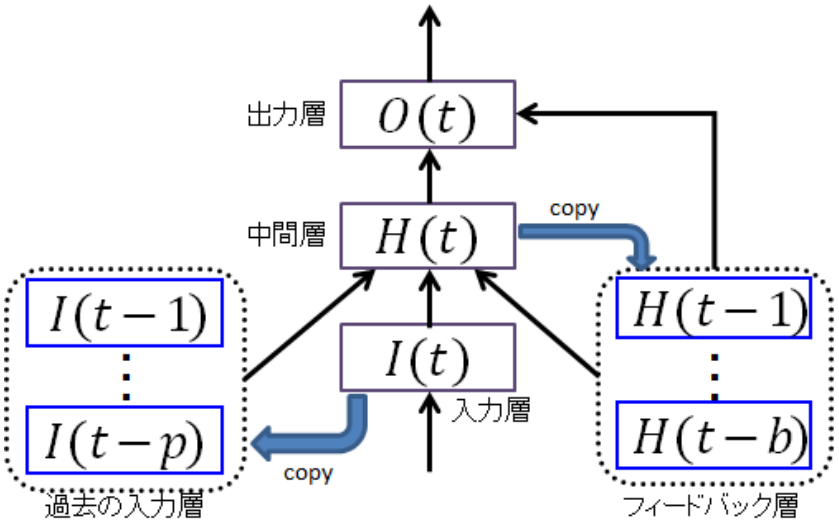

図 3.9: extended MCRNN (exMCRNN)[27]

3.2.6. Long-Short Term Memoy (LSTM)

Long-Short Term Memory ネットワークは，長期的な時系列特性を学習することのできる RNN の一種であり，Hochreiter and Schmidhuber[6] により提案された．LSTM は 3 つのゲートとメモリを用いることによってデータの忘却，更新，出力を行い，長期的な時系列特性の学習が困難な問題を回避するように設計されている．exMCRNN では「一次遅れ＋むだ時間」系のように長期的な時系列特性を持つ場合

がある制御対象の入出力データを用いて学習を行った際に，古い情報が消失し学習することが困難な場合が存在したが，LSTM では，セルの内部において情報を長期的に保存するための「メモリ」と「入力ゲート」,「忘却ゲート」,「出力ゲート」の 3 つのゲートを設けることで，古い情報を消失させず保持でき，長期間の時系列特性を適切に学習可能なモデルとなっている．LSTM の構造を図 3.10 に示す．

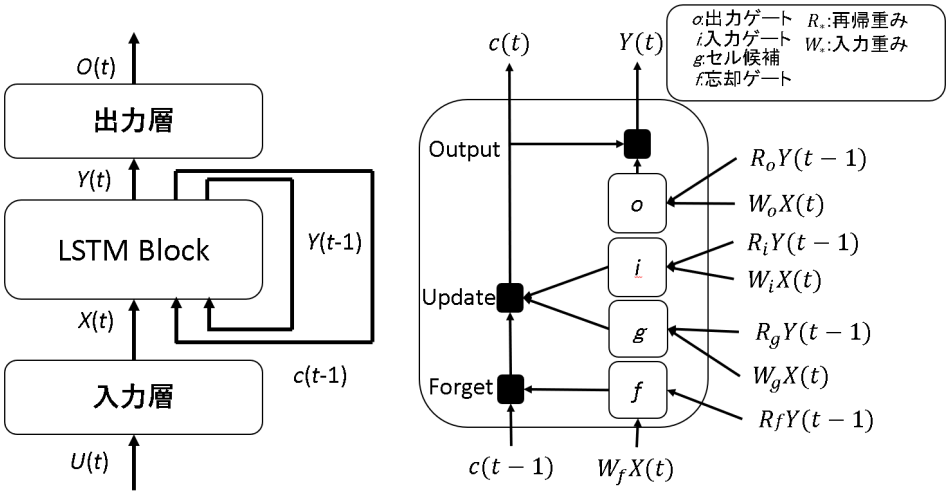

図 3.10: Long-Short Term Memory (LSTM)[6]

LSTM の各ゲートでの働きについて説明する．LSTM の忘却ゲートでは，長い期間保持している時系列情報から重要度の低い情報を削除する働きを行っている．現在の入力と一つ前のユニットからの出力値に基づいて，情報の保持割合を計算することで不必要な情報の忘却を行う．次に，入力ゲートで通常の RNN と同様に，入力と一つ前のセルからの出力値を用いてセルの状態を計算し，必要な情報だけをメモリセルに保持する．一つ前のメモリセルからの情報に対して，適宜に不必要な情報を忘却し，新しく追加される時系列データの情報に基づいて計算される時系列特性を考慮して，最新のメモリセルの状態に更新を行う．最後に出力ゲートで入力値と一つ前のセルからの出力値に基づき，長期記憶に保存されている情報と出力ゲートからの出力の二つのデータを出力する．以上の過程を行うことのより時系列データの忘却，更新，出力を行う．

LSTM 層では，入力重み $\boldsymbol{W}$(Input Weights)，再帰重み $\boldsymbol{R}$(Recurrent Weights)，およびバイアス $\boldsymbol{b}$(Bias) の 3 種類の重みの学習を行う．これらの重みとバイアスの行列

は，式 (3.4) のように連結し定義される．

$$\boldsymbol{W} = \begin{bmatrix} W_i \\ W_f \\ W_g \\ W_o \end{bmatrix}, \boldsymbol{R} = \begin{bmatrix} R_i \\ R_f \\ R_g \\ R_o \end{bmatrix}, \boldsymbol{b} = \begin{bmatrix} b_i \\ b_f \\ b_g \\ b_o \end{bmatrix} \tag{3.4}$$

ここで，添え字で書かれている i は入力ゲート，f は忘却ゲート，g はセル候補，および o は入力ゲートを表す．LSTM ではセルの長期の時系列情報を保つための変数 $\boldsymbol{c_t}$ を設定している．長期記憶 $\boldsymbol{c_t}$ に対して，古い情報と新しい情報を適宜追加・忘却することで，適当な長期の時系列情報の保持を可能にしている．次に新しい入力情報と現在の状態をもとに算出された新しい情報をメモリセルに追加する．そして，新たに得られた長期の時系列情報を次のセルに伝播する．

LSTM の忘却ゲートでは，入力 $\boldsymbol{X_t}$ と一つ前のユニットからの出力値 $\boldsymbol{Y_{t-1}}$ を用いて，記憶率 $\boldsymbol{f_t}$ を算出する．タイムステップ t での忘却ゲートは式 (3.5) で与えられる．

$$\boldsymbol{f_t} = \sigma_g(W_f \boldsymbol{X_t} + R_f \boldsymbol{Y_{t-1}} + b_f) \tag{3.5}$$

このとき，σ_g はシグモイド関数 $\sigma(x) = (1 + e^{-x})^{-1}$ を用いて決定されるゲート活性化関数である．$\boldsymbol{f_t}$ は，一期前のセルの長期記憶 $\boldsymbol{c_t}$ にかかるため，$\boldsymbol{c_t}$ をどの程度保持するのかを決定する係数と考えることができる．

次に，入力 $\boldsymbol{X_t}$ と一期前のセルからの出力値 $\boldsymbol{Y_{t-1}}$ から，現在のセルの状態 $\boldsymbol{g_t}$ を計算する．時刻 t でのセル候補は式 (3.6) で求められる．

$$\boldsymbol{g_t} = \sigma_c(W_g \boldsymbol{X_t} + R_g \boldsymbol{Y_{t-1}} + b_g) \tag{3.6}$$

このとき，σ_c はセルの状態を決定する活性化関数を表し，双曲線正弦関数 (tanh) を用いてセルの状態の活性化関数を計算する．また，時刻 t での入力ゲートは式 (3.7) で与えられる．

$$\boldsymbol{i_t} = \sigma_g(W_i \boldsymbol{X_t} + R_i \boldsymbol{Y_{t-1}} + b_i) \tag{3.7}$$

次に，一期前のセルが保持する長期の時系列情報 $\boldsymbol{c_{t-1}}$ に対して，不要な情報を削除し，現在の入力値を用いて計算される情報を付加して，最新の長期の時系列情報 $\boldsymbol{c_t}$ とする．時刻 t での長期の時系列情報は式 (3.8) で与えられる．

$$\boldsymbol{c_t} = \boldsymbol{f_t} \odot \boldsymbol{c_{t-1}} + \boldsymbol{i_t} \odot \boldsymbol{g_t} \tag{3.8}$$

ここで，$\odot$ は行列の要素ごとの乗算を表す．時刻 t のセルからの出力値は，入力情報 X_t と一期前の出力値 Y_{t-1} をもとにして，メモリセルに保持されいる長期の時系列情報を加えて、出力値を求める．このとき，出力ゲートと LSTMblock からの出力

は次式で与えられる．

$$\boldsymbol{o_t} = \sigma_g(W_o\boldsymbol{X_t} + R_o\boldsymbol{Y_{t-1}} + b_o) \tag{3.9}$$

$$\boldsymbol{Y_t} = \boldsymbol{o_t} \odot +\sigma_c(\boldsymbol{c_t}) \tag{3.10}$$

3.3. ニューラルネットワークの学習

ニューラルネットワークは，学習により適切な結合重みを算出する必要がある．ニューラルネットワークの学習方法には，2種類の学習方法があり，教師なし学習と教師あり学習の2種類に大別できる．教師なし学習では，教師信号を事前に用意し与えず，モデルの設計者自身が設けた評価基準をもとに学習を行う．教師なし学習の代表的な例として，コホーネンの学習アルゴリズムが挙げられる [28]．一方，教師あり学習では，入力データに対する理想の出力値が教師信号として与えられ，ニューラルネットワークからの出力と教師信号との差を最小化するように結合重みの値を決定する．教師あり学習の代表的な例としては，誤差逆伝搬法 (backpropagation) がある．

3.3.1. 誤差逆伝搬法 (back propagation)

誤差逆伝搬法 (back propagation) は，1986年に Rumelhart *et al.*[29] によって提案された教師あり学習アルゴリズムの1種であり，最も幅広く利用されている学習アルゴリズムの1つである．

対象とするニューラルネットワークの，第t番目の入出力パターンに対する第$(s-1)$層第iニューロンの出力を$x_i^{s-1}(t)$，第s層第jニューロンの出力を$x_j^s(t)$とすると，ニューロンの入出力の関係は次式のようになる．

$$x_j^s(t) = f_j\left(\sum_i w_{ij}x_i^{s-1}(t) - \theta_j\right) \tag{3.11}$$

ここで，$f_j(\cdot)$は第s層第jニューロンに対応する活性化関数であり，w_{ij}は第$(s-1)$層第iニューロンと第s層第jニューロンとを結ぶ結合の重み，θ_jは第s層第jニューロンの閾値である．

第t期目の入出力パターンに対応する，出力層の第kユニットからの出力値を$O_k(t)$とし，第kユニットに対応する教師信号を$d_k(t)$とする．このとき，第t番目の入出力パターンに対する誤差関数と，すべての入出力パターンに対する総誤差関数は次

式で定義される．

$$E(t) = \frac{1}{2}\sum_{k}(d_k(t) - O_k(t))^2 \tag{3.12}$$

$$E = \sum_{t}^{T} E(t) \tag{3.13}$$

ここで，T は入出力パターンの総数である．

誤差逆伝搬法は最急降下法の一種であり，ニューラルネットワークの総誤差 E を最小化するために，各入出力パターンに対する誤差関数の勾配を計算し，誤差が最小化されるように重みを更新する．このとき，式 (3.11) における重み w_{ij} の変化量は次式のように与えられる．

$$\Delta w_{ij} = -\alpha\frac{\partial E(t)}{\partial w_{ij}} \tag{3.14}$$

ここで，α は w_{ij} の変化の大きさを表す係数であり，一般に学習率と呼ばれる．学習率は，大きいほど学習速度が速くなるが極小値の付近で振動したり発散するなど上手く収束しなくなるため，適切な値に設定する必要がある．出力層での誤差関数が式 (3.13) であり，シグモイド関数によって表現される活性化関数を用いたとき，ネットワークの層の数を S，第 $s-1$ 層のユニット i から第 s 層のユニット j への結合重みを w_{ij}^s で表される．具体的な更新の方法は式 (3.15) で表される．

$$\Delta w_{ij}^s = -\alpha\delta_i^s O_j^{s-1} \tag{3.15}$$

ここで，O_j^{s-1} は第 $s-1$ 層のユニット j の出力値である．

式 (3.15) に含まれる δ_i^s は合成関数の微分法に従い，出力層での二乗誤差 E から計算される δ_i^S を用いて，再帰的に計算される．

$$\delta_i^S = -(d_i - O_i^S)O_i'^S \tag{3.16}$$

$$\delta_i^{s-1} = (\sum_{j=1}\delta_i^s w_{ij}^s)O_i'^{s-1} \tag{3.17}$$

ここで，$O_i'^s$ は第 s 層ユニット i の出力値の微分値である．この計算より，出力層での誤差を前の層に伝播させてゆく形になっていることが分かる．誤差逆伝播法では，総誤差 E の値が十分小さくなったところで学習を終了する．

3.3.2. Back Propagation Through Time

再帰構造を持つ RNN は BackPropagation through time（BPTT）によって学習することで，巨大な Deep Neural Network(DNN) の学習とみなすことができるため，誤差逆伝播法と同様に最適化をすることができる [30]．本項では，図 3.6 に示す RNN を考え，$I_i(t)$ を入力層ユニット i からの出力値，$H_j(t)$ を中間層ユニット j からの出

力値，$O_k(t)$ を出力層ユニット k からの出力値，v_{ij} を入力ユニット i から中間層ユニット j への重み，u_{lj} をフィードバック層 (過去の中間層) ユニット l から中間層ユニット j への重み，w_{jk} を中間層ユニット j から出力層ユニット k への重みとする．

誤差逆伝播法と同様に各層の δ を計算するために，出力層ユニット k の $\delta_k^o(t)$ は以下のように計算する．

$$\delta_k^o(t) = -(d_k(t) - O_k(t))O_k'(t) \tag{3.18}$$

t 期における中間層ユニット j は，t 期の出力層ユニットと $t+1$ 期の中間層ユニットと接続されているので，$\delta_j^h(t)$ は以下のように計算できる．

$$\delta_j^h(t) = (\sum_k w_{jk}\delta_k^o(t) + \sum_l u_{jl}\delta_l^h(t+1))H_j'(t) \tag{3.19}$$

各時刻における出力層の $\delta_k^o(t)$ が計算できれば，各時刻における中間層の $\delta_j^o(t)$ が計算できる．各時刻 $t = 1, \cdots T$ における出力層の $\delta_k^o(t)$ が与えられれば，$t = T$ から始め，t を 1 ずつ小さくしながら，式 (3.19) を繰り返し計算することで，各時刻における中間層の $\delta_j^h(t)$ が計算できる．ただし，この時点で，$t = T+1$ における $\delta_j^h(T+1)$ は計算できないため，$\delta_j^h(T+1) = 0$ とする．

各時刻，各層の δ が計算できたら，誤差 E の各層の重みによる微分は次のように計算できる．

$$\frac{\partial E}{\partial v_{ij}} = \sum_{t=1}^{T} \delta_j^h(t) I_i(t) \tag{3.20}$$

$$\frac{\partial E}{\partial u_{lj}} = \sum_{t=1}^{T} \delta_j^h(t) H_j(t-1) \tag{3.21}$$

$$\frac{\partial E}{\partial w_{jk}} = \sum_{t=1}^{T} \delta_j^h(t) H_j(t) \tag{3.22}$$

式 (3.20),(3.21), (3.22) を式 (3.14) に代入することで，各層の重みの調整をすることができる．

3.3.3. 1step ごとにデータを与える場合の学習

本論文では 2 種類の学習方法について提案する．1step ごとにデータを与える場合の LSTM の学習には (3.23) 式で定義される平均二乗誤差 E を誤差関数として用いる．

$$E = \frac{1}{2} \cdot \frac{1}{T} \sum_{t=1}^{T} \sum_{m=1}^{M} (d_m(t) - O_m(t))^2 \tag{3.23}$$

このとき，T はステップ数，M は出力ユニット数すなわち変動のクラス数，$d_m(t)$ は教師信号，$O_m(t)$ は LSTM からの出力値である．この誤差関数を前述の誤差逆伝播法によって学習用データを用いて教師信号と LSTM からの出力値の誤差を最小化する方向に調整する．

3.3.4. 複数 step ごとにデータを与える場合の学習

複数 step ごとにデータを与える場合の LSTM の学習には (3.24) 式で定義される平均二乗誤差 E を誤差関数として用いる．

$$E = \frac{1}{2}\sum_{m=1}^{M}(d_m - O_m)^2 \tag{3.24}$$

このとき，M は出力ユニット数すなわち変動のクラス数，d_m は教師信号の出力値，O_m は LSTM の出力ユニットからの出力値である．また，教師信号は One-hot で表現され，各信号に対応するラベルを付与する．この誤差関数を前述の誤差逆伝播法によって学習用データを用いて教師信号と LSTM からの出力値の誤差を最小化する方向に調整する．

3.4. NN を用いたシステム変動分類

本論文の目的は，制御対象にシステム変動が起きた場合に，どのパラメータがどの程度変動したのかを RNN の出力 $\boldsymbol{O}(t)$ から予測・分類することである，制御対象のシステム変動を適切に分類することができれば，それに基づいてコントローラを設定することで，制御性能を向上させることである．図 3.11 に提案手法の概念図を示す．

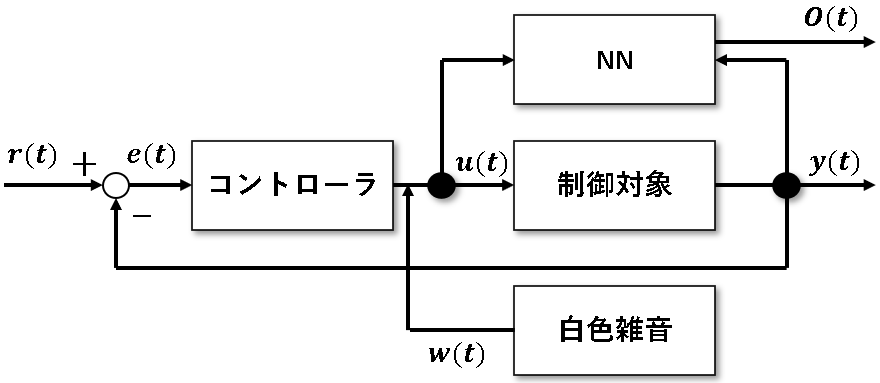

図 3.11: 提案手法のブロック線図

本論文では，システム変動の種類を分類するために，システム変動を起こしていないデータと，いくつかのシステム変動後のデータを用いて学習した RNN の出力に基づいて分類を行う．

3.4.1. 1step ごとにデータを与える場合のシステム変動分類

制御対象からの入出力データ $(\Delta u(t-1), y(t), \Delta y(t))$ を LSTM の入力信号とし，LSTM の出力ユニット数は分類するクラス数に基づいて決定される．

出力層の m 番目の出力ユニット $O_m(t)$ は以下の式によって計算される．

$$O_m(t) = \sigma_c \left(\sum_{l=1}^{h} w_{lm} Y_l(t) \right) \tag{3.25}$$

ここで，h は中間ユニット数であり，w は中間層のユニットと出力層のユニットの結合重みである．重みは学習用データを用いて調整する．NN からの出力値 $\boldsymbol{O}(t)$ は式 (3.26) で表される．

$$\boldsymbol{O}(t) = [O_1(t), O_2(t), \cdots, O_m(t)] \quad (t = 1, 2, ..., T) \tag{3.26}$$

出力値 $\boldsymbol{O}(t)$ を各クラスごとに平均 $\bar{\boldsymbol{O}}$ をとることで，式 (3.27) のように出力値を計算する．

$$\begin{aligned} \bar{\boldsymbol{O}} &= \left[\frac{\sum_{t=1}^{T} O_1(t)}{T}, \frac{\sum_{t=1}^{T} O_2(t)}{T}, \cdots, \frac{\sum_{t=1}^{T} O_M(t)}{T} \right] \\ &= [\bar{O}_1, \bar{O}_2, \cdots, \bar{O}_M] \end{aligned} \tag{3.27}$$

このとき，平均出力値 $\bar{\boldsymbol{O}}$ から平均出力値の差分を計算し，平均出力値の差分と閾値を用いてシステム変動の種類を分類する．

3.4.2. 複数 step ごとにデータを与える場合のシステム変動分類

出力値の差分に基づいて算出した場合と同様に，制御対象の入出力データ $(\Delta u(t-1), y(t), \Delta y(t))$ を LSTM の入力信号とし，LSTM の出力ユニット数は分類するクラス数に基づいて決定される．

出力層の m 番目の出力ユニット O_m は以下の式によって計算される．

$$O_m = \sigma_c \left(\sum_{l=1}^{h} w_{lm} Y_l(t) \right) \tag{3.28}$$

ここで，h は中間ユニット数であり，w は中間層のユニットと出力層のユニットの結合重みである．重みは学習用データを用いて調整する．NN からの出力値 $\boldsymbol{O}$ は式 (3.29) で表される．

$$\boldsymbol{O} = [O_1, O_2, \cdots, O_m] \quad (m = 1, 2, ..., M) \tag{3.29}$$

上記の出力値 $\boldsymbol{O}$ と教師信号 $\boldsymbol{d}$ の比較を行い，対応するクラスに分類する．

3.5. 多段階モデルを用いたシステム変動分類

本論文では，多段階モデルを用いたシステム変動分類手法を行う．図 3.12 に多段階モデルの概念図を示す．一段階目の分類結果をもとに似た特徴を持つと考えられる入出力データを同一のグループとして，各グループのデータのみを用いて分類を行い，これを第二段階とする．同様に学習データがすべて分類できるまで多段階に分けて分類する．似た特徴を持つデータごとにグループ分けを行いグループごとに学習を行うことによって，よりわずかな特徴の差に対応したを学習を行うことができるようになり，一段階では分類することのできない特徴を検出することができるようになると考えられる．

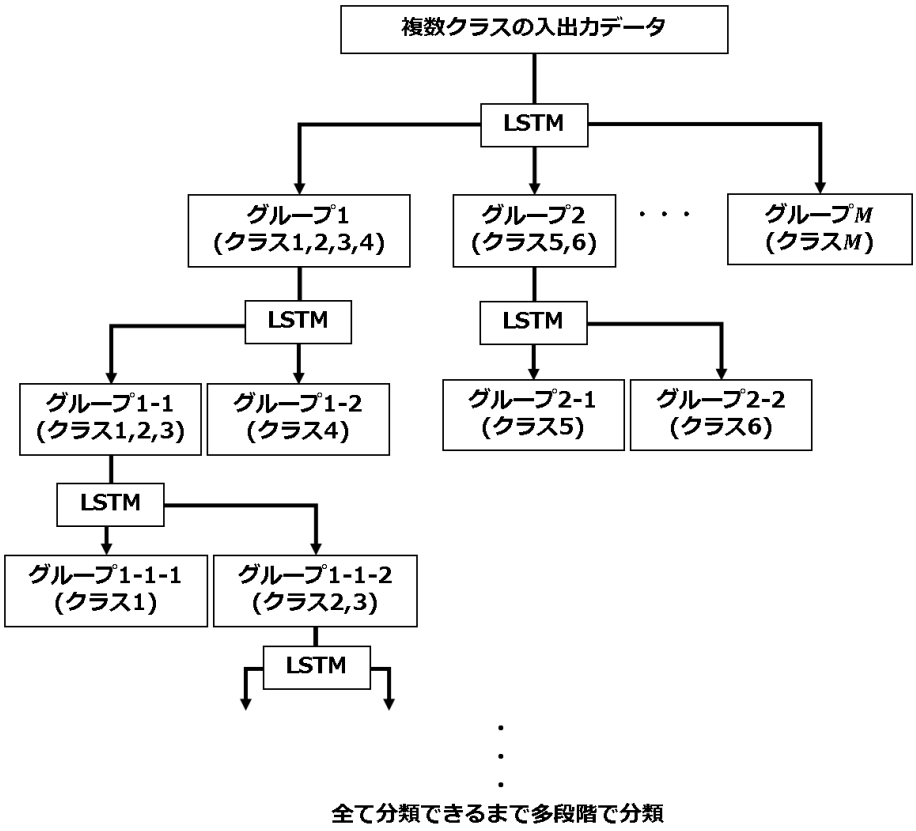

図 3.12: 多段階モデルの概念図

第 4 章

数値実験

本論文では，システムゲイン K，時定数 T，むだ時間 L をそれぞれ変動させた実験と K と T，K と L，T と L をそれぞれ同時に変動させた実験を行い，それぞれ K 変動実験，T 変動実験，L 変動実験，KT 変動実験，KL 変動実験，TL 変動実験とする．分類器は前述の LSTM を用いる．なお，LSTM の中間層数を 100，学習率を 0.001，学習回数を 10000 回とし，学習回数が 10000 回に到達するか，もしくは，学習用データの分類精度が 100%になった時点で学習を終了させる．また，本論文では学習用データ，過学習が起きていないか確認するための確認用データ，学習したモデルの汎化能力を確認するための検証用データの 3 種類の制御対象への 1000step 分の入出力データ $(\Delta u(t-1), y(t), \Delta y(t))$ をそれぞれ 100 サンプルずつ用意し学習・検証を行う．

4.1. サンプルデータの生成

本実験では次式で表される「一次遅れ＋むだ時間」系の制御対象を考える．

$$G(s) = \frac{10}{1+100s} e^{-8s} \tag{4.1}$$

(4.1) 式をサンプリング時間 $T_s = 1$[s] で離散化すると，(2.2) 式のむだ時間 $d = 8$ となり，$A(z^{-1}), B(z^{-1})$ は以下のようになる．

$$\left.\begin{aligned} A(z^{-1}) &= 1 - 0.990z^{-1} \\ B(z^{-1}) &= 0.0995z^{-1} \end{aligned}\right\} \tag{4.2}$$

ただし，平均 0，分散 0.003^2 のガウス性白色雑音を制御対象への入力として付加する．PID ゲインは木下ら [16] の手法を用いて，以下のように求めた．

$$K_P = 0.28, K_I = 0.01, K_D = 1.80 \tag{4.3}$$

また，システム変動は 5001–7500[step] で起こるものとし，2001–3000[step] における $(\Delta u(t-1), y(t), \Delta y(t))$ を変動前のデータとして，7501–8500[step] における $(\Delta u(t-1), y(t), \Delta y(t))$ を変動後のデータとして LSTM への入力信号に用いる．

4.2. システムゲイン K の変動分類実験

K 変動実験のシステム変動は以下の変動式で表わされる．

$$K = \begin{cases} 10 & (t \leqq 5000) \\ 10 + \frac{\Delta K(t-5000)}{2500} & (5000 < t \leqq 7500) \\ 10 + \Delta K & (t > 7500) \end{cases} \tag{4.4}$$

ΔK はシステム変動の大きさを表しており，本実験では ΔK に $-7, 7$ のいずれかを代入し，学習用データとして用いた．変動後に $K < 10$ のデータをクラス K^-，変動していない $K = 10$ のデータをクラス K^0，$K > 10$ のデータをクラス K^+ とし，各クラスに対応するデータを 100 サンプルずつ取得して学習を行う．取得したデータの一例を図 4.1,4.2 に示す．

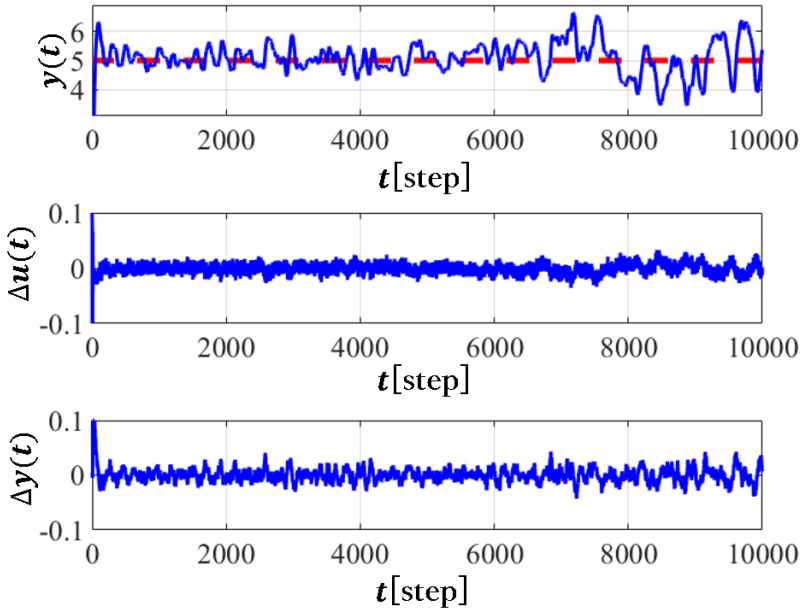

図 4.1: $\Delta K = -7 (K = 3)$ の制御結果 (クラス K^-)

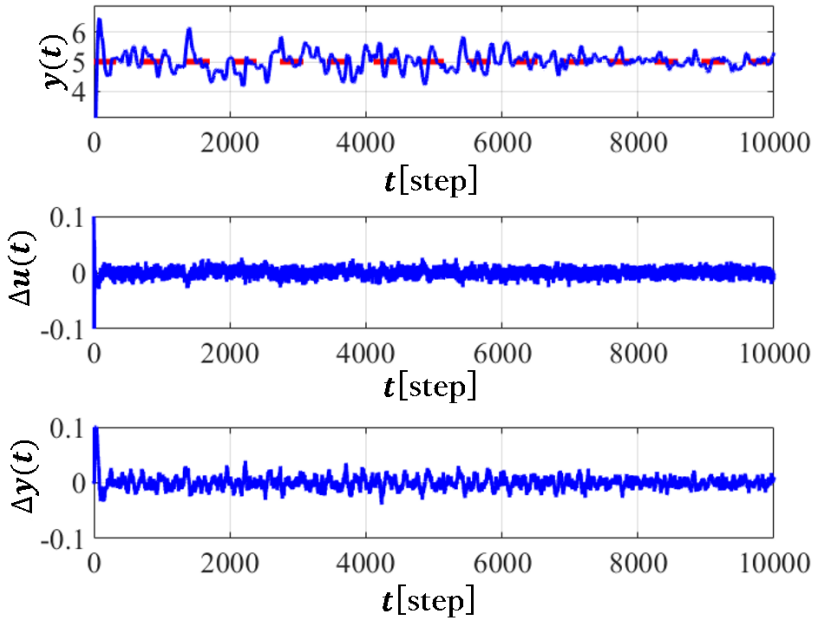

図 4.2: $\Delta K = 7 (K = 17)$ の制御結果 (クラス K^{+})

本論文では 2 種類の変動分類実験を行う．1step ごとにデータを与えて学習を行う変動分類実験では，教師信号としてクラス K^{-} に属するデータに対しては $\boldsymbol{O}_m^K = [1, -1]$，同様にクラス K^0 に属するデータに対しては $\boldsymbol{O}_m^K = [0, 0]$，クラス K^{+} に属するデータに対しては $\boldsymbol{O}_m^K = [-1, 1]$ を与える．

複数 step ごとにデータを与えて学習を行う変動分類実験では，教師信号としてクラス K^{-} に属するデータに対しては $\boldsymbol{O}_m^K = [1, 0, 0]$，同様にクラス K^0 は $\boldsymbol{O}_m^K = [0, 1, 0]$，クラス K^{+} は $\boldsymbol{O}_m^K = [0, 0, 1]$ を与える．

複数 step ごとにデータを与えて学習を行う変動分類実験と 1step ごとにデータを与えて学習を行う変動分類実験における学習用データ各 100 サンプルずつの分類結果を表 4.1 にまとめる．縦軸は学習に用いたデータのシステム変動の大きさを表しており，横軸は各クラスに分類されたサンプル数を示している．

表 4.1: システムゲイン K の変動実験における学習用データの分類結果

	1step ごとに学習する場合			複数 step ごとに学習する場合		
	クラス K^-	クラス K^0	クラス K^+	クラス K^-	クラス K^0	クラス K^+
$\Delta K = -7$	100	0	0	100	0	0
$\Delta K = 0$	0	100	0	0	37	63
$\Delta K = 7$	0	0	100	0	26	74

表 4.1 より複数 step ごとにデータを与えて学習を行った場合はいずれの場合もシステム変動の種類に対応するクラスに分類することができており，システム変動の特徴を適切に学習できていることが分かる．一方で，1step ごとにデータを与えて学習を行った場合はクラス K^0，クラス K^+ に対応するクラスの分類精度が悪くなっており，クラス K^0，クラス K^+ の入出力データを適切に学習することが難しいことが分かる．また，計算時間に関して，1step ごとにデータを与えて学習を行った場合に比べて複数 step ごとにデータを与えて学習を行った場合の方が 0.05 倍程度に計算時間を短縮することができ，より効率的に時系列の特徴を学習することができると考えられる．

次に各変動分類実験において学習を行った LSTM を用いて未知の検証用データの分類を行う．

複数 step ごとにデータを与えて学習を行う変動分類実験と 1step ごとにデータを与えて学習を行う変動分類実験における検証用データ各 100 サンプルずつの分類結果を表 4.2 にまとめる．

表 4.2: LSTM を用いた K 変動実験分類結果

	複数 step ごとに学習する場合			1step ごとに学習する場合		
	クラス K^-	クラス K^0	クラス K^+	クラス K^-	クラス K^0	クラス K^+
$\Delta K = -7$	100	0	0	100	0	0
$\Delta K = -6$	89	11	0	97	3	0
$\Delta K = -5$	78	21	1	80	19	1
$\Delta K = -3$	26	73	1	25	48	27
$\Delta K = -2$	22	76	2	8	55	37
$\Delta K = -1$	14	80	6	4	53	43
$\boldsymbol{\Delta K = 0}$	**14**	**77**	**9**	**0**	**37**	**63**
$\Delta K = 1$	5	81	14	0	25	75
$\Delta K = 2$	4	62	34	0	29	71
$\Delta K = 3$	1	45	54	0	32	68
$\Delta K = 5$	2	30	68	0	27	73
$\Delta K = 6$	0	9	91	0	17	83
$\Delta K = 7$	0	0	100	0	26	74

複数 step ごとにデータを与えて学習を行った場合の分類結果と 1step ごとにデータを与えて学習を行う分類結果を比較すると，1step ごとにデータを与えて学習を行った場合では分類されるデータがクラス K^+ に偏っているのに対して，複数 step ごとにデータを与えて学習を行った場合の方が変動の種類に対応したクラスに分類される割合が多くなっている，この結果から複数 step ごとにデータを与えて学習を行った場合の方がより特徴を捉えられていると考えられる．一方で，ΔK が -3〜2 の範囲ではクラス K^0 に分類されているサンプルの割合が多くなっている．この結果から微小な変動を分類することが困難な場合が存在していると考えられる．

4.3. 時定数 T の変動分類実験

T 変動実験のシステム変動は以下の変動式で表わされる。

$$T = \begin{cases} 100 & (t \leqq 5000) \\ 100 + \frac{\Delta T(t-5000)}{2500} & (5000 < t \leqq 7500) \\ 100 + \Delta T & (t > 7500) \end{cases} \tag{4.5}$$

ΔT はシステム変動の大きさを表しており，本実験では ΔT に $-60, 60$ のいずれかを代入し，学習用データとして用いた．ここで，変動後に $T < 100$ のデータをクラス T^- とし，変動していない $T = 100$ のデータをクラス T^0，$T > 100$ のデータをクラス T^+ とし，各クラスに対応するデータを 100 サンプルずつ取得して学習を行う．取得したデータの一例を図 4.3,4.4 に示す．

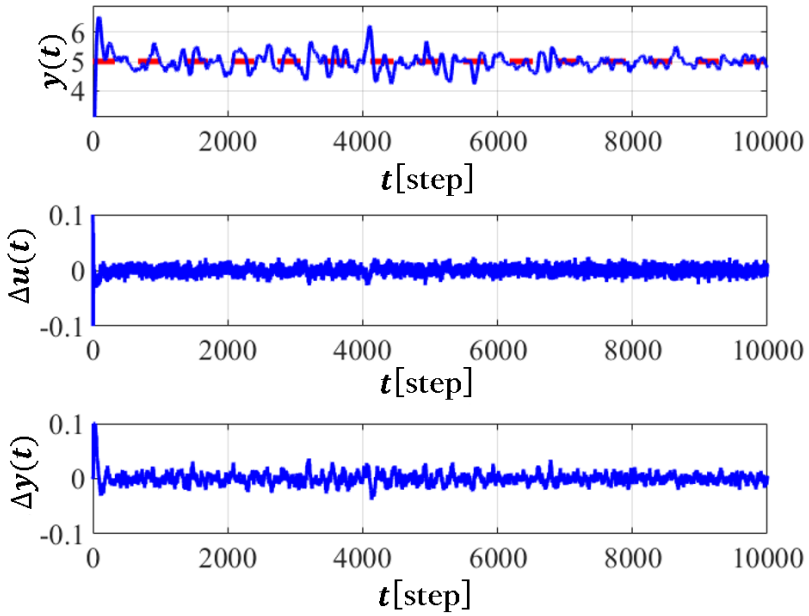

図 4.3: $\Delta T = -60 (T = 40)$ の制御結果 (クラス T^-)

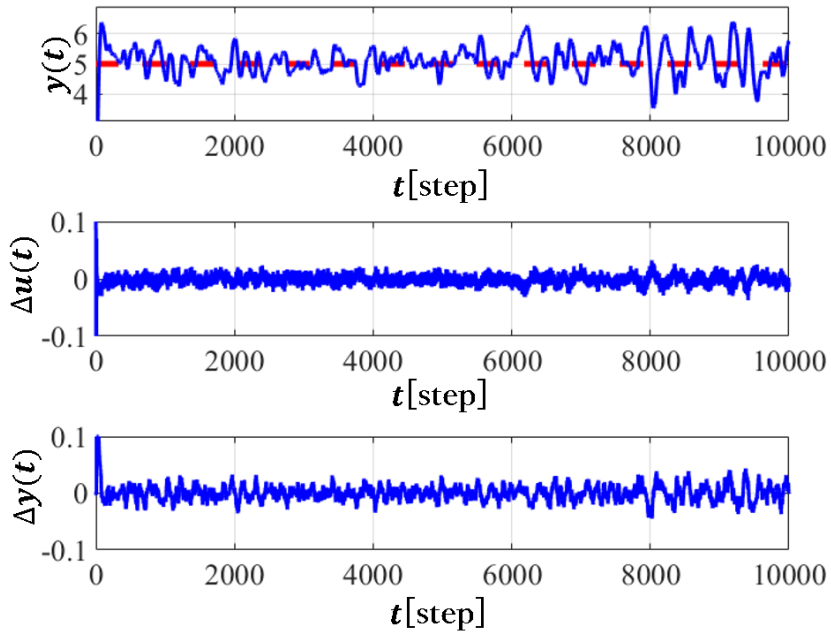

図 4.4: $\Delta T = 60 (T = 160)$ の制御結果 (クラス T^{+})

K 変動実験と同様に 2 種類の変動分類実験を行う．1step ごとにデータを与えて学習を行う変動分類実験では，教師信号としてクラス T^{-} に属するデータに対しては $\boldsymbol{O}_m^T = [1, -1]$，同様にクラス T^0 に属するデータに対しては $\boldsymbol{O}_m^T = [0, 0]$，クラス T^{+} に属するデータに対しては $\boldsymbol{O}_m^T = [-1, 1]$ を与える．

複数 step ごとにデータを与えて学習を行う変動分類実験では，教師信号としてクラス T^{-} に属するデータに対しては $\boldsymbol{O}_m^T = [1, 0, 0]$，同様にクラス T^0 は $\boldsymbol{O}_m^T = [0, 1, 0]$，クラス T^{+} は $\boldsymbol{O}_m^T = [0, 0, 1]$ を与える．

複数 step ごとにデータを与えて学習を行う変動分類実験と 1step ごとにデータを与えて学習を行う変動分類実験における学習用データ各 100 サンプルずつの分類結果を表 4.3 にまとめる．縦軸は学習に用いたデータのシステム変動の大きさを表しており，横軸は各クラスに分類されたサンプル数を示している．

表 4.3: 時定数 T の変動実験における学習用データの分類結果

	1step ごとに学習する場合			複数 step ごとに学習する場合		
	クラス T^-	クラス T^0	クラス T^+	クラス T^-	クラス T^0	クラス T^+
$\Delta T = -60$	100	0	0	100	0	0
$\Delta T = 0$	0	100	0	39	23	38
$\Delta T = 60$	0	0	100	5	17	78

表 4.3 より複数 step ごとにデータを与えて学習を行った場合はいずれの場合もシステム変動の種類に対応するクラスに分類することができており，システム変動の特徴を適切に学習できていることが分かる．一方で，1step ごとにデータを与えて学習を行った場合はクラス T^0，クラス T^+ に対応するクラスの分類精度が悪くなっており，クラス T^0，クラス T^+ の入出力データを適切に学習することが難しいことが分かる．

次に各変動分類実験において学習を行った LSTM を用いて未知の検証用データの分類を行う．

複数 step ごとにデータを与えて学習を行う変動分類実験と 1step ごとにデータを与えて学習を行う変動分類実験における検証用データ各 100 サンプルずつの分類結果を表 4.4 にまとめる．

表 4.4: LSTM を用いた T 変動実験分類結果

	複数 step ごとに学習する場合			1step ごとに学習する場合		
	クラス T^-	クラス T^0	クラス T^+	クラス T^-	クラス T^0	クラス T^+
$\Delta T = -60$	100	0	0	100	0	0
$\Delta T = -50$	87	12	1	92	5	3
$\Delta T = -40$	42	56	2	89	6	5
$\Delta T = -30$	15	80	5	64	17	19
$\Delta T = -20$	4	88	8	59	19	22
$\Delta T = -10$	2	83	15	42	22	36
$\boldsymbol{\Delta T = 0}$	**0**	**79**	**21**	**39**	**23**	**38**
$\Delta T = 10$	0	60	40	29	21	50
$\Delta T = 20$	0	49	51	22	21	57
$\Delta T = 30$	0	46	54	11	27	62
$\Delta T = 40$	0	39	61	16	29	55
$\Delta T = 50$	0	21	79	16	23	61
$\Delta T = 60$	0	0	100	5	25	70

表 4.4 より，1step ごとにデータを与えて学習を行った場合では分類されるデータがクラス T^- に偏っているのに対して，複数 step ごとにデータを与えて学習を行った場合の方が変動の種類に対応したクラスに分類される割合が多くなっている，この結果から複数 step ごとにデータを与えて学習を行った場合の方がより特徴を捉えられていると考えられる．一方で，ΔT が -30〜10 の範囲ではクラス K^0 に分類されているサンプルの割合が多くなっている．この結果から K と同様に微小な変動を分類することが困難な場合が存在していると考えられる．

4.4. むだ時間 L の変動分類実験

L 変動実験のシステム変動は以下の変動式で表わされる。

$$L = \begin{cases} 8 & (t \leqq 5000) \\ 8 + \frac{\Delta L(t-5000)}{2500} & (5000 < t \leqq 7500) \\ 8 + \Delta L & (t > 7500) \end{cases} \tag{4.6}$$

ΔL はシステム変動の大きさを表しており，本実験では ΔL に $-6, 20$ のいずれかを代入し，学習用データとして用いた．ここで，変動後に $L < 8$ のデータをクラス L^- とし，変動していない $L = 8$ のデータをクラス L^0，$L > 8$ のデータをクラス L^+ とし，各クラスに対応するデータを 5 サンプルずつ取得して学習を行う．取得したデータの一例を図 4.5,4.6 に示す．

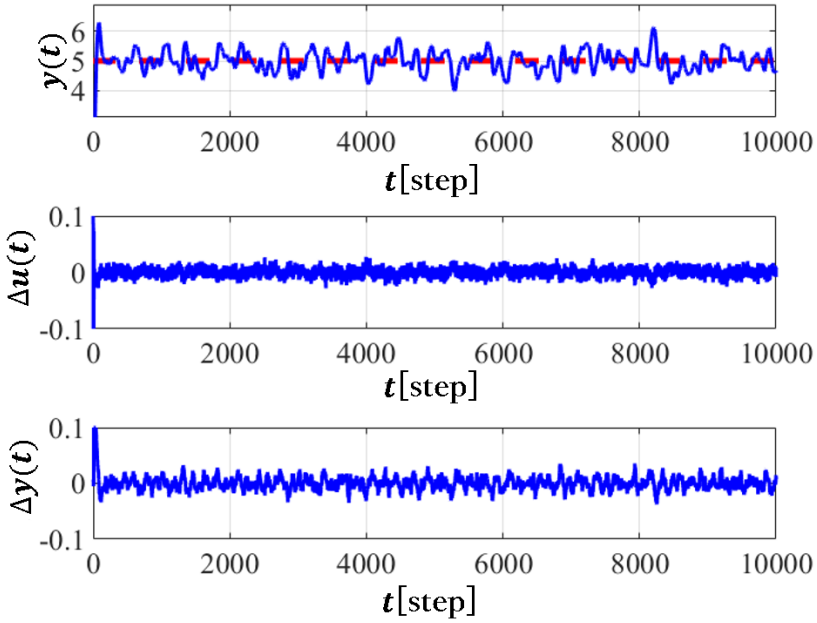

図 4.5: $\Delta L = -6 (L = 2)$ の制御結果 (クラス L^-)

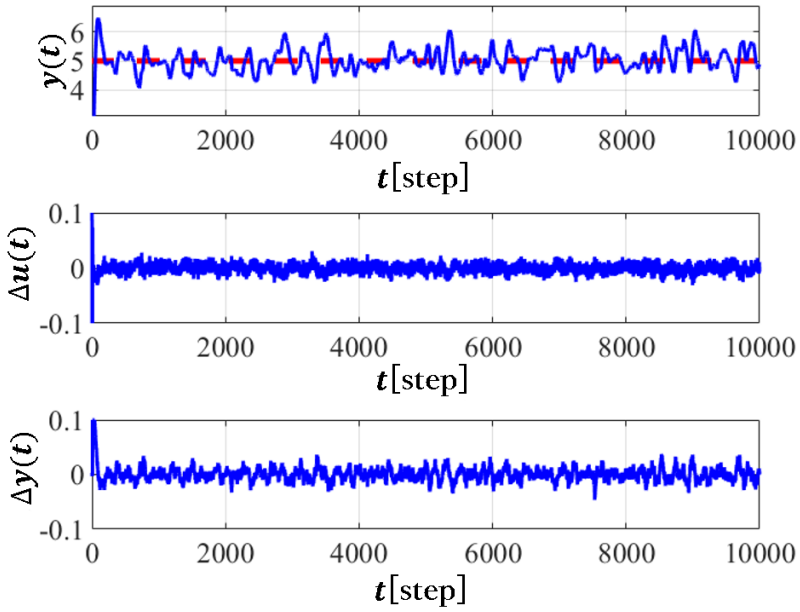

図 4.6: $\Delta L = 20(L = 28)$ の制御結果 (クラス L^{+})

K,T 同様に 2 種類の変動分類実験を行う．1step ごとにデータを与えて学習を行う変動分類実験では，教師信号としてクラス L^{-} に属するデータに対しては $\boldsymbol{O}_m^L = [1, -1]$，同様にクラス L^0 に属するデータに対しては $\boldsymbol{O}_m^L = [0, 0]$，クラス L^{+} に属するデータに対しては $\boldsymbol{O}_m^L = [-1, 1]$ を与える．

複数 step ごとにデータを与えて学習を行う変動分類実験では，教師信号としてクラス L^{-} に属するデータに対しては $\boldsymbol{O}_m^L = [1, 0, 0]$，同様にクラス L^0 は $\boldsymbol{O}_m^L = [0, 1, 0]$，クラス L^{+} は $\boldsymbol{O}_m^L = [0, 0, 1]$ を与える．

複数 step ごとにデータを与えて学習を行う変動分類実験と 1step ごとにデータを与えて学習を行う変動分類実験における学習用データ各 100 サンプルずつの分類結果を表 4.5 にまとめる．縦軸は学習に用いたデータのシステム変動の大きさを表しており，横軸は各クラスに分類されたサンプル数を示している．

表 4.5: むだ時間 L の変動実験における学習用データの分類結果

	1step ごとに学習する場合			複数 step ごとに学習する場合		
	クラス L^-	クラス L^0	クラス L^+	クラス L^-	クラス L^0	クラス L^+
$\Delta L = -6$	100	0	0	72	26	2
$\Delta L = 0$	0	100	0	49	42	9
$\Delta L = 20$	0	0	100	0	4	96

表 4.5 より複数 step ごとにデータを与えて学習を行った場合はいずれの場合もシステム変動の種類に対応するクラスに分類することができており，システム変動の特徴を適切に学習できていることが分かる．一方で，1step ごとにデータを与えて学習を行った場合はクラス L^-，クラス L^0 に対応するクラスの分類精度が悪くなっており，クラス L^-，クラス L^0 の入出力データを適切に学習することが難しいことが分かる．

次に各変動分類実験において学習を行った LSTM を用いて未知の検証用データの分類を行う．

複数 step ごとにデータを与えて学習を行う変動分類実験と 1step ごとにデータを与えて学習を行う変動分類実験における学習用データ各 100 サンプルずつの分類結果を表 4.6 にまとめる．

表 4.6: LSTM を用いた L 変動実験分類結果

	1step ごとに学習する場合			複数 step ごとに学習する場合		
	クラス L^-	クラス L^0	クラス L^+	クラス L^-	クラス L^0	クラス L^+
$\Delta L = -6$	100	0	0	72	26	2
$\Delta L = -5$	54	46	0	71	24	5
$\Delta L = -4$	40	60	0	72	26	2
$\Delta L = -2$	45	55	0	64	32	4
$\Delta L = -1$	46	52	2	58	37	5
$\boldsymbol{\Delta L = 0}$	**38**	**58**	**4**	**49**	**42**	**9**
$\Delta L = 2$	36	62	2	43	43	14
$\Delta L = 4$	33	62	5	35	47	18
$\Delta L = 10$	19	45	36	3	54	43
$\Delta L = 12$	12	26	62	4	46	50
$\Delta L = 20$	0	0	100	0	4	96

表 4.6 より，1step ごとにデータを与えて学習を行った場合では分類されるデータがクラス L^- に偏っているのに対して，複数 step ごとにデータを与えて学習を行った場合の方が変動の種類に対応したクラスに分類される割合が多くなっている．この結果から複数 step ごとにデータを与えて学習を行った場合の方がより特徴を捉えられていると考えられる．一方で，ΔL が -5〜10 の範囲ではクラス L^0 に分類されているサンプルの割合が多くなっている．この結果から K，T と同様に微小な変動を分類することが困難な場合が存在していると考えられる．

4.5. K, T が同時に変動する場合の変動分類実験

K, T 変動実験のシステム変動は以下の変動式で表わされる．

$$K = \begin{cases} 10 & (t \leqq 5000) \\ 10 + \frac{\Delta K(t-5000)}{2500} & (5000 < t \leqq 7500) \\ 10 + \Delta K & (t > 7500) \end{cases} \tag{4.7}$$

$$T = \begin{cases} 100 & (t \leqq 5000) \\ 100 + \frac{\Delta T(t-5000)}{2500} & (5000 < t \leqq 7500) \\ 100 + \Delta T & (t > 7500) \end{cases} \tag{4.8}$$

ΔK，ΔT はそれぞれシステム変動の大きさを表しており，本実験では ΔK に -7,, 7，ΔT に -60，60 のいずれかを代入し，それぞれ学習用データとして用いた．変動後に $K < 10$ のデータをクラス K^-，変動していない $K = 0$ のデータをクラス K^0，$K > 10$ のデータをクラス K^+ とし，変動後に $T < 100$ のデータをクラス T^-，変動していない $T = 100$ のデータをクラス T^0，$T > 100$ のデータをクラス T^+ として同時に変動させ，9 種類のクラスにそれぞれ (K^*, T^*)，$* \in \{0, +, -\}$ としてラベルを与える．各クラスに対応するデータを 100 サンプルずつ取得して学習を行う．取得したデータの一例を図 4.7–4.14 に示す．

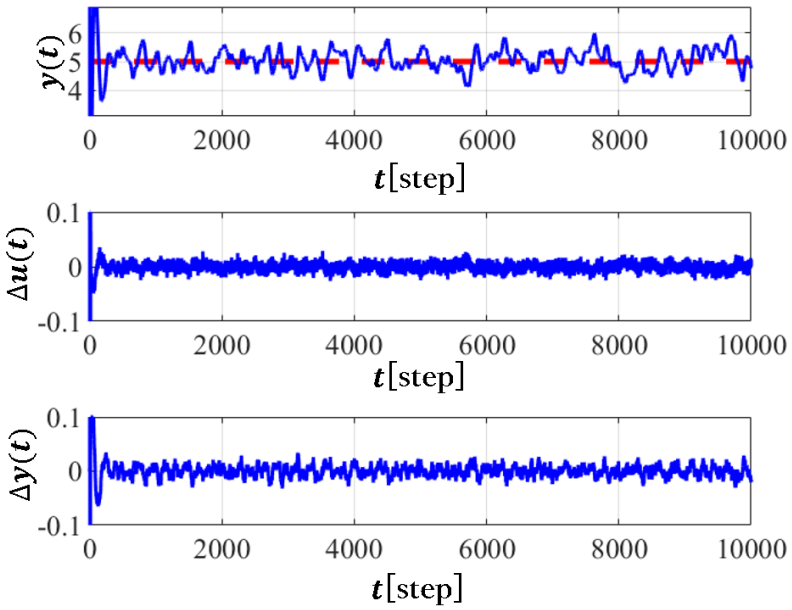

図 4.7: $\Delta K = -7(K = 3), \Delta T = -60(T = 40)$ の制御結果 (クラス K^-, T^-)

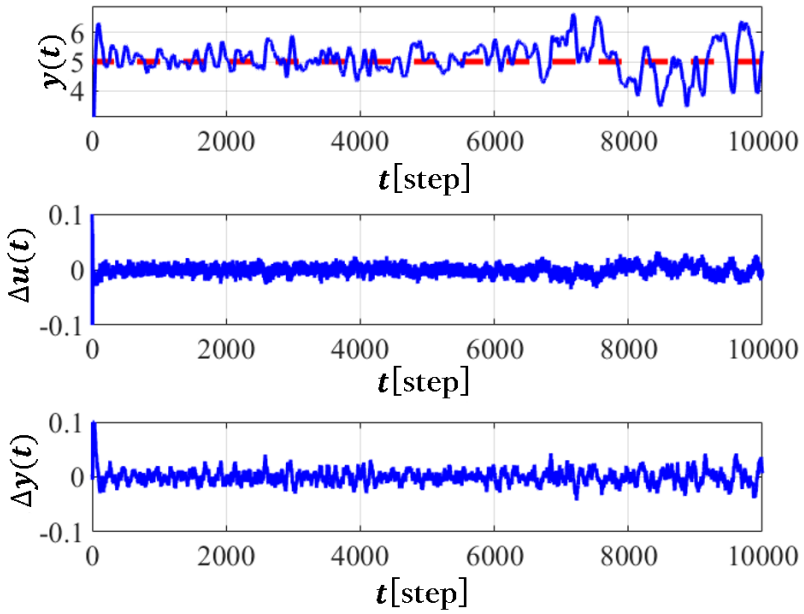

図 4.8: $\Delta K = -7(K = 3), \Delta T = 0(T = 100)$ の制御結果 (クラス K^-, T^0)

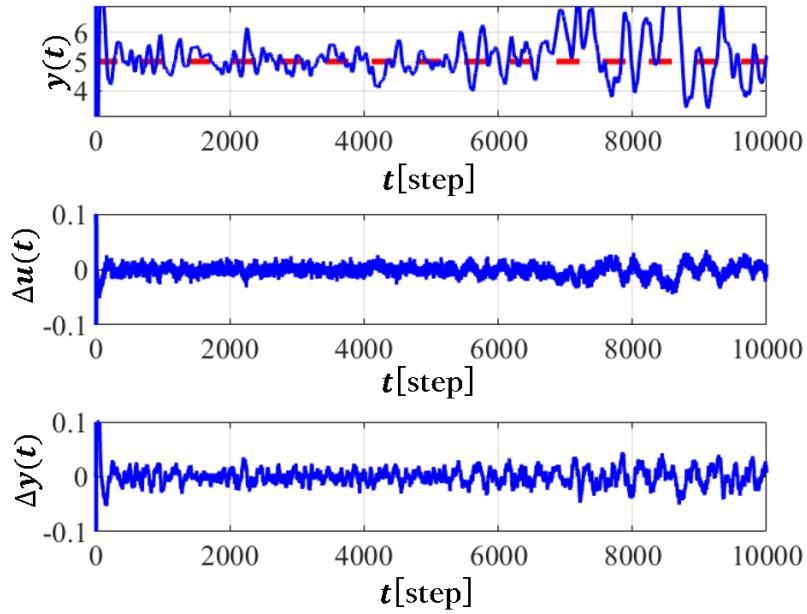

図 4.9: $\Delta K = -7(K = 3), \Delta T = 60(T = 160)$ の制御結果 (クラス K^-, T^+)

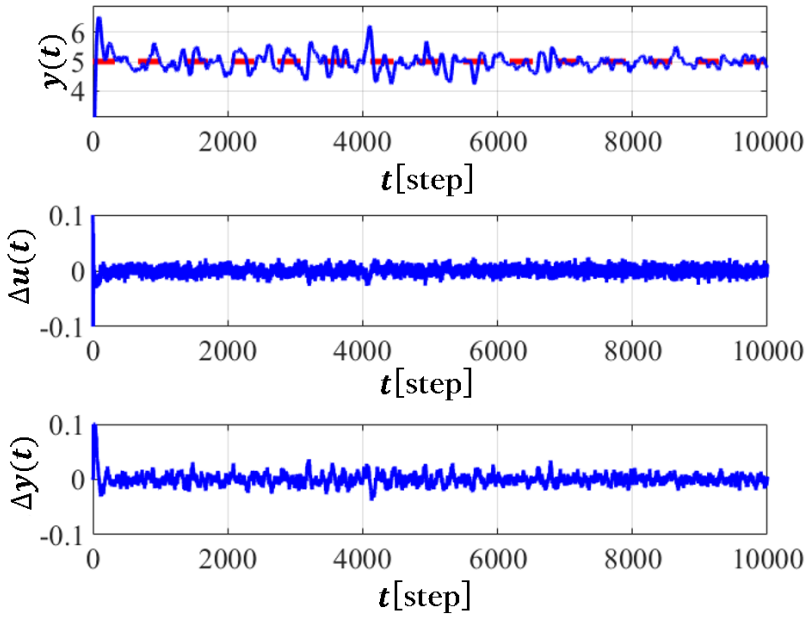

図 4.10: $\Delta K = 0(K = 10), \Delta T = -60(T = 40)$ の制御結果 (クラス K^0, T^-)

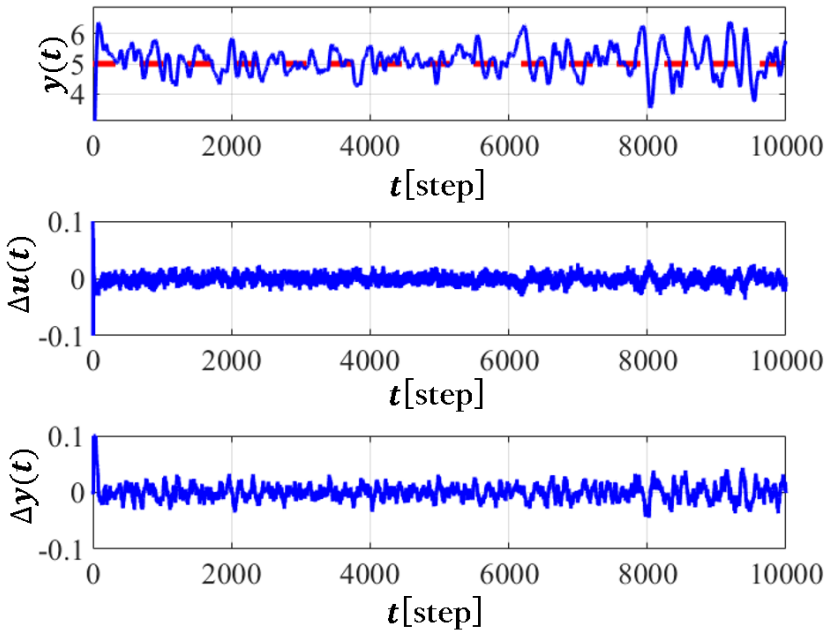

図 4.11: $\Delta K = 0(K = 10), \Delta T = 60(T = 160)$ の制御結果 (クラス K^0, T^+)

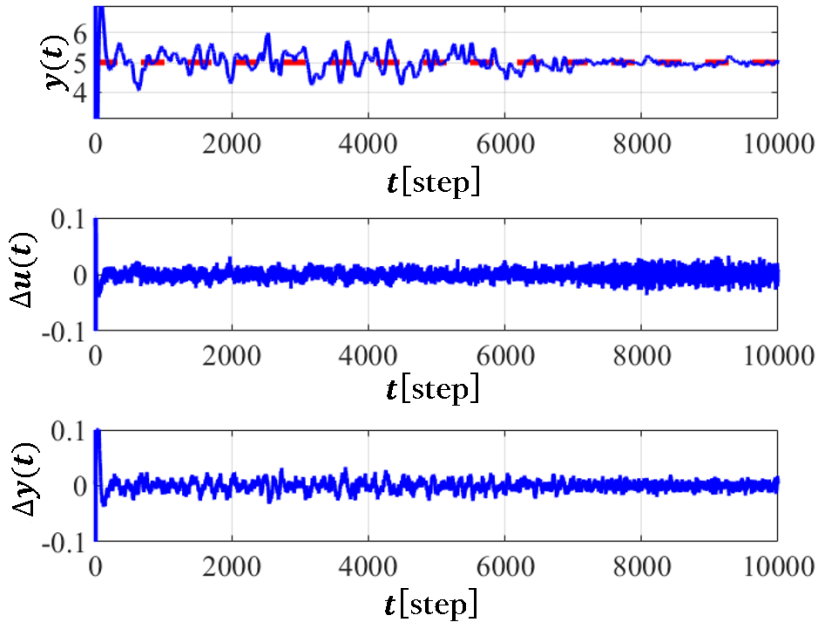

図 4.12: $\Delta K = 7(K = 17), \Delta T = -60(T = 40)$ の制御結果 (クラス K^+, T^-)

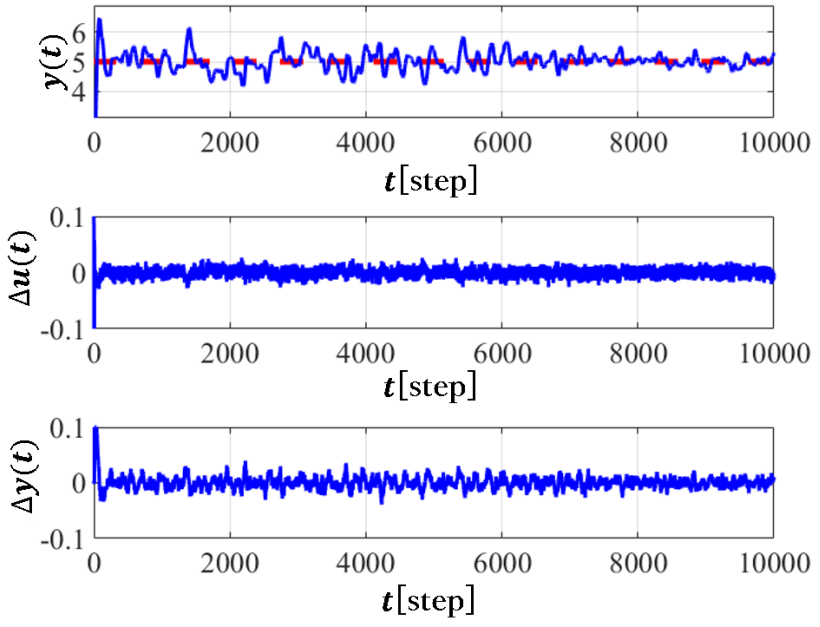

図 4.13: $\Delta K = 7(K = 3), \Delta T = 0(T = 100)$ の制御結果 (クラス K^+, T^0)

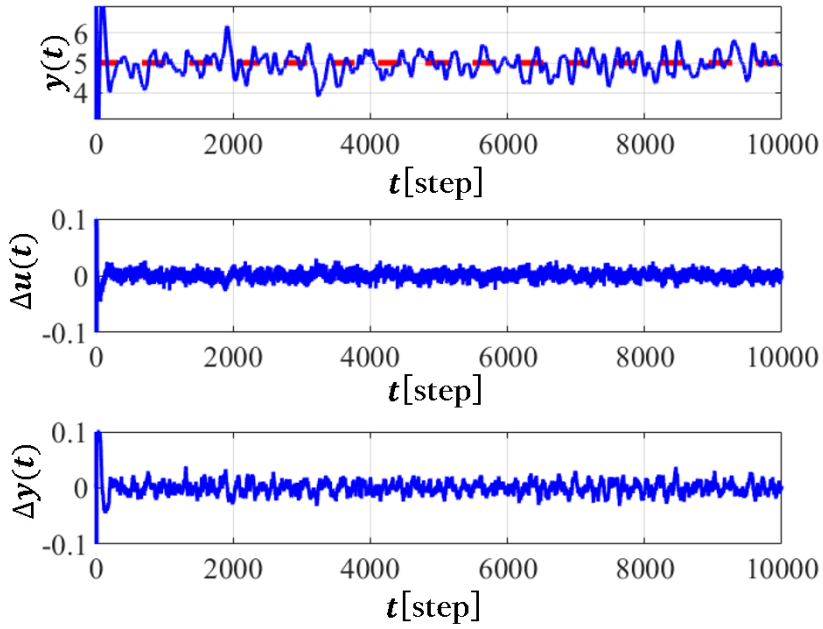

図 4.14: $\Delta K = 7(K = 17), \Delta T = 60(T = 160)$ の制御結果 (クラス K^+, T^+)

K, T が同時に変動する場合の変動分類実験では，複数 step ごとにデータを与えて学習を行う．また，LSTM の出力を O_m^{KT} とし，教師信号は One-hot 表現でそれぞれ与えられる．

KT が同時に変動する場合の変動分類実験における一段階目の学習用データ各 100 サンプルずつの分類結果を表 4.7 にまとめる．行は実際に分類されたデータのクラスを表しており，列は学習に用いたデータのサンプル数を示している．

表 4.7: KT 変動分類実験における一段階目の学習用データの分類結果

クラス	K^-, T^-	K^-, T^0	K^-, T^+	K^+, T^+	K^0, T^0	K^0, T^+	K^0, T^-	K^+, T^0	K^+, T^-
K^-, T^-	**100**	0	0	0	0	0	0	0	0
K^-, T^0	0	**100**	0	0	0	0	0	0	0
K^-, T^+	0	0	**100**	0	0	0	0	0	0
K^+, T^+	0	0	0	**100**	0	0	0	0	0
K^0, T^0	0	0	0	0	**100**	0	0	0	0
K^0, T^+	0	0	0	0	0	**100**	0	0	0
K^0, T^-	0	0	0	0	0	0	**100**	0	0
K^+, T^0	0	0	0	0	0	0	0	**100**	0
K^+, T^-	0	0	0	0	0	0	0	0	**100**

表 4.7 から，変動の特徴に関係したクラスに 100%の精度で分類されており，正しく学習が完了していると考えられる．

次に学習を行った LSTM を用いて未知の検証用データの分類を行う．

KT が同時に変動する場合の変動分類実験における検証用データ各 100 サンプルずつの分類結果を表 4.8 にまとめる．

表 4.8: KT 変動分類実験における一段階目の検証用データの分類結果

クラス	K^-,T^-	K^-,T^0	K^-,T^+	K^+,T^+	K^0,T^0	K^0,T^+	K^0,T^-	K^+,T^0	K^+,T^-
K^-,T^-	**25**	9	6	15	19	12	5	18	0
K^-,T^0	4	**22**	23	5	1	14	0	0	0
K^-,T^+	6	28	**35**	9	10	16	0	3	0
K^+,T^+	23	9	5	**22**	32	22	4	16	0
K^0,T^0	12	8	6	18	**17**	6	17	0	0
K^0,T^+	12	20	24	12	5	**25**	0	3	0
K^0,T^-	5	0	0	4	1	1	**60**	17	3
K^+,T^0	13	4	1	15	15	3	19	**25**	0
K^+,T^-	0	0	0	0	0	0	6	1	**93**

学習用データと比較して変動の特徴に関係したクラスに分類されるデータの割合が少なくなっており，クラスの分類に偏りが生じているものが存在した．この結果から正しいクラスに分類されたデータ数に対して他のクラスに誤分類され比率が大きくなっているデータは似た特徴を持ち分類が困難になっていると考えられる．

表 4.9 に一段階目の検証用データの分類結果をもとにグループ分けを行った結果をまとめる．表 4.9 をもとに各グループの検証用データを用いて第二段階の学習・分類実験を行う．

表 4.9: 一段階目の分類結果をもとにしたグループ分け

	クラス			
グループ 1	K^-, T^-	K^-, T^0	K^-, T^+	K^+, T^+
グループ 2	K^0, T^0	K^0, T^+		
グループ 3	K^0, T^-	K^+, T^0		
グループ 4	K^+, T^-			

KT 変動分類実験における二段階目の学習用データ各 100 サンプルずつの分類結果を表 4.10-4.12 にまとめる.

表 4.10: KT 変動分類実験における二段階目の学習用データの分類結果 (グループ 1)

クラス	K^-, T^-	K^-, T^0	K^-, T^+	K^+, T^+
K^-, T^-	**100**	0	0	0
K^-, T^0	0	**100**	0	0
K^-, T^+	0	0	**100**	0
K^+, T^+	0	0	0	**100**

表 4.11: KT 変動分類実験における二段階目の学習用データの分類結果 (グループ 2)

クラス	K^0, T^0	K^0, T^+
K^0, T^0	**100**	0
K^0, T^+	0	**100**

表 4.12: KT 変動分類実験における二段階目の学習用データの分類結果 (グループ 3)

クラス	K^0, T^-	K^+, T^0
K^0, T^-	**100**	0
K^+, T^0	0	**100**

表 4.10-4.12 から，変動の特徴に関係したクラスに 100%の精度で分類されており，正しく学習が完了していると考えられる.

次に学習を行った LSTM を用いて各グループに対して二段階目の未知の検証用データの分類を行う.

KT が同時に変動する場合の変動分類実験における二段階目の検証用データ各 100 サンプルずつの分類結果を表 4.13-4.15 にまとめる．

表 4.13: KT 変動分類実験における二段階目の検証用データの分類結果 (グループ 1)

クラス	K^-, T^-	K^-, T^0	K^-, T^+	K^+, T^+
K^-, T^-	**45**	5	1	42
K^-, T^0	5	**42**	29	1
K^-, T^+	5	44	**70**	3
K^+, T^+	45	9	0	**54**

表 4.14: KT 変動分類実験における二段階目の検証用データの分類結果 (グループ 2)

クラス	K^0, T^0	K^0, T^+
K^0, T^0	**72**	32
K^0, T^+	28	**68**

表 4.15: KT 変動分類実験における二段階目の検証用データの分類結果 (グループ 3)

クラス	K^0, T^-	K^+, T^0
K^0, T^-	**93**	10
K^+, T^0	7	**90**

表 4.13 に関して，変動の特徴に関係したクラスに分類されるデータの割合が少なくなっており，クラスの分類に偏りが生じているものが存在した．分類結果から (K^-, T^-) と (K^+, T^+)，(K^-, T^0) と (K^-, T^+) が似た特徴を持っていると考えられ，再度グループ分けを行い三段階目の学習及び変動分類実験を行う必要があると考えられる．表 4.14 に関して，約 70%の精度で正しいクラスに分類することができている．より精度を高めるためには信号処理などの手法を用いることで特徴量を明確化したうえでの学習・分類が必要であると考えられる．表 4.14 に関しては，90%以上の精度で正しいクラスに分類することができており，十分に適切に分類することができていると考えられる．

表 4.16 に二段階目の検証用データの分類結果をもとにグループ分けを行った結果をまとめる．表 4.16 をもとに各グループの検証用データを用いて第三段階の学習・分類実験を行う．

表 4.16: 二段階目の分類結果をもとにしたグループ分け

	クラス	
グループ 1-1	K^-, T^-	K^+, T^+
グループ 1-2	K^-, T^0	K^-, T^+
グループ 2	K^0, T^0	K^0, T^+
グループ 3	K^0, T^-	K^+, T^0
グループ 4	K^+, T^-	

KT が同時に変動する場合の変動分類実験における三段階目の学習用データ各 100 サンプルずつの分類結果を表 4.17，4.18 にまとめる．

表 4.17: KT 変動分類実験における三段階目の学習用データの分類結果 (グループ 1-1)

クラス	K^0, T^0	K^0, T^+
K^0, T^0	**100**	0
K^0, T^+	0	**100**

表 4.18: KT 変動分類実験における三段階目の学習用データの分類結果 (グループ 1-2)

クラス	K^0, T^-	K^+, T^0
K^0, T^-	**100**	0
K^+, T^0	0	**100**

表 4.17-4.18 から，変動の特徴に関係したクラスに 100%の精度で分類されており，正しく学習が完了していると考えられる．

次に学習を行った LSTM を用いて各グループに対して三段階目の未知の検証用データの分類を行う．

KT が同時に変動する場合の変動分類実験における三段階目の検証用データ各 100 サンプルずつの分類結果を表 4.19，4.20 にまとめる．

表 4.19: KT 変動分類実験における三段階目の検証用データの分類結果 (グループ 1-1)

クラス	K^-, T^-	K^+, T^+
K^-, T^-	**56**	50
K^+, T^+	44	**50**

表 4.20: KT 変動分類実験における三段階目の検証用データの分類結果 (グループ 1-2)

クラス	K^-, T^0	K^-, T^+
K^-, T^0	**50**	29
K^-, T^+	50	**71**

表 4.19 に関してどちらのクラスも分類精度が 50%程度になっており適切に分類することができていない．表 4.20 に関しても K^-, T^+ のクラスの入出力データに関しては約 70%の精度で正しいクラスに分類することができている一方で K^-, T^0 のクラスに関しては分類精度が 50%となっており適切に分類することができていない．以上の結果からより分類精度を高めるためには信号処理などの手法を用いることで特徴量を明確化したうえでの学習・分類が必要であると考えられる．

4.6. K, L が同時に変動する場合の変動分類実験

K, L 変動実験のシステム変動は以下の変動式で表わされる．

$$K = \begin{cases} 10 & (t \leqq 5000) \\ 10 + \frac{\Delta K(t-5000)}{2500} & (5000 < t \leqq 7500) \\ 10 + \Delta K & (t > 7500) \end{cases} \tag{4.9}$$

$$L = \begin{cases} 8 & (t \leqq 5000) \\ 8 + \frac{\Delta L(t-5000)}{2500} & (5000 < t \leqq 7500) \\ 8 + \Delta L & (t > 7500) \end{cases} \tag{4.10}$$

ΔK，ΔL はそれぞれシステム変動の大きさを表しており，本実験では ΔK に -7,, 7，ΔL に -6，6 のいずれかを代入し，それぞれ学習用データとして用いた．変動後に $K < 10$ のデータをクラス K^-，変動していない $K = 0$ のデータをクラス K^0，$K > 10$ のデータをクラス K^+ とし，変動後に $L < 8$ のデータをクラス L^-，変動していない $L = 8$ のデータをクラス L^0，$L > 8$ のデータをクラス L^+ として同時に変動させ，9 種類のクラスにそれぞれ (K^*, L^*)，$* \in \{0, +, -\}$ としてラベルを与える．各クラスに対応するデータを 100 サンプルずつ取得して学習を行う．取得したデータの一例を図 4.15–4.22 に示す．

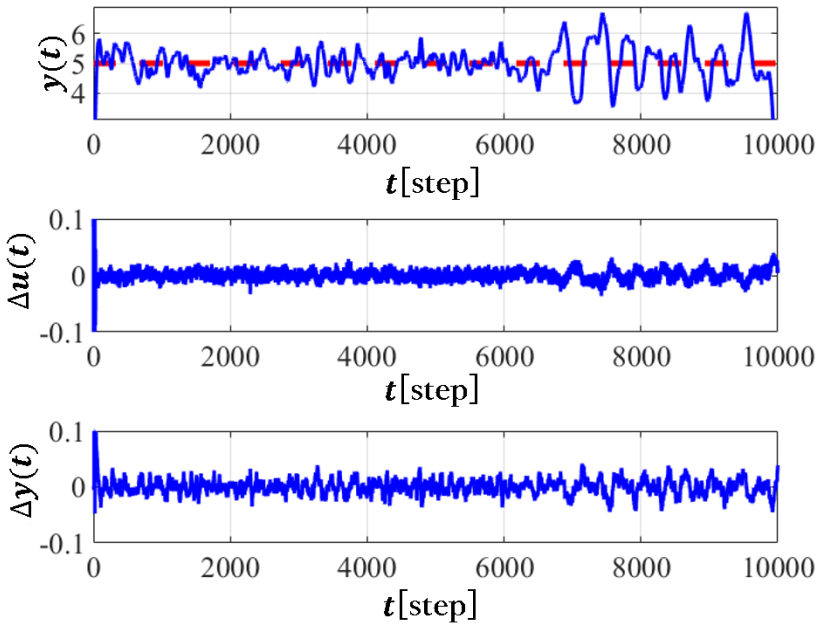

図 4.15: $\Delta K = -7(K = 3), \Delta L = -6(L = 2)$ の制御結果 (クラス K^-, L^-)

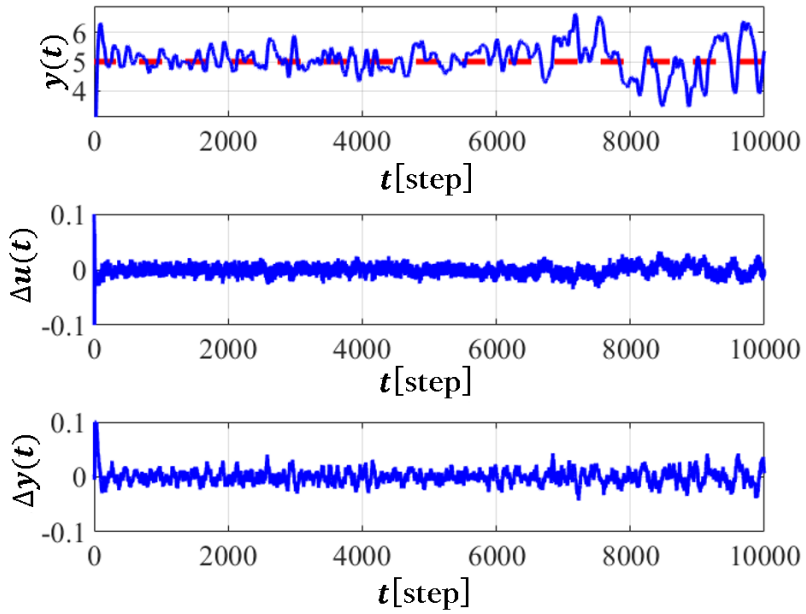

図 4.16: $\Delta K = -7(K = 3), \Delta L = 0(L = 8)$ の制御結果 (クラス K^-, L^0)

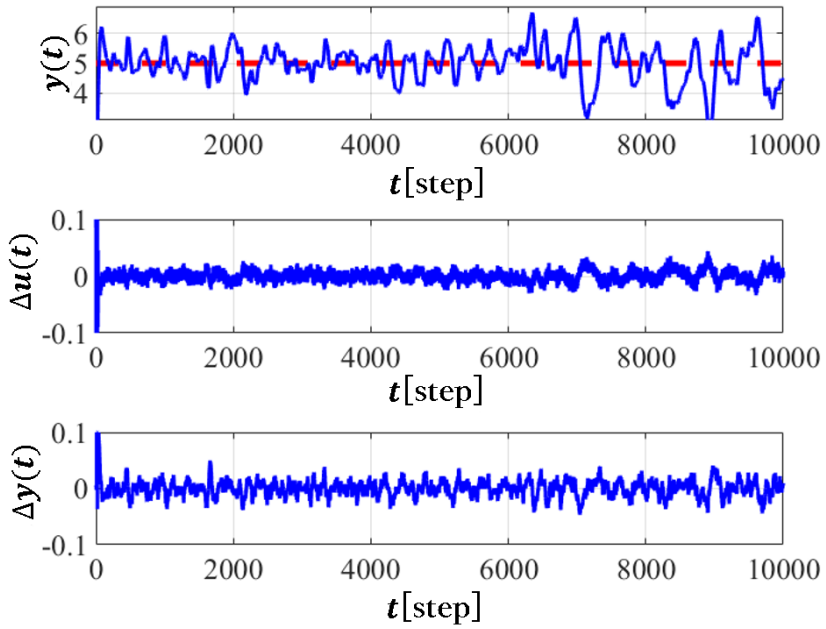

図 4.17: $\Delta K = -7(K = 3), \Delta L = 6(L = 14)$ の制御結果 (クラス K^-, L^+)

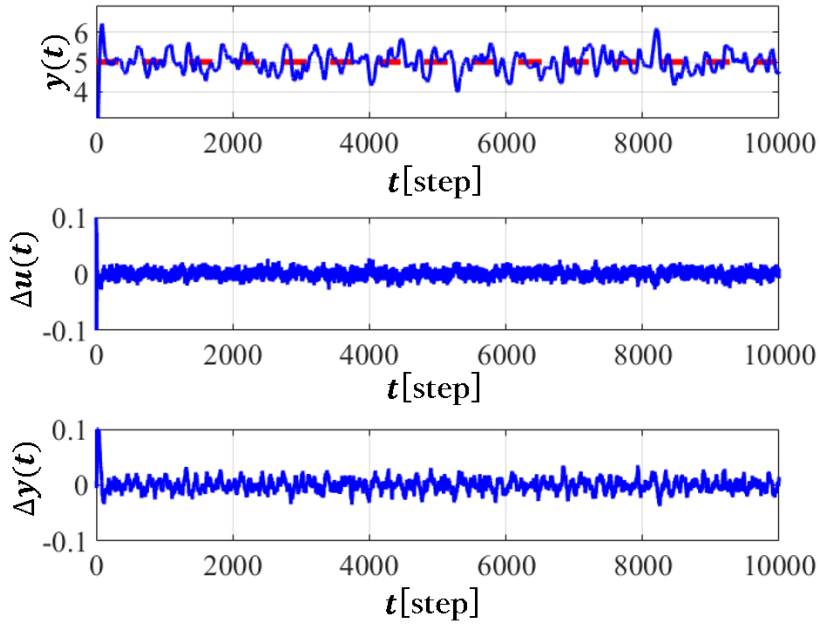

図 4.18: $\Delta K = 0(K = 10), \Delta L = -6(L = 2)$ の制御結果 (クラス K^0, L^-)

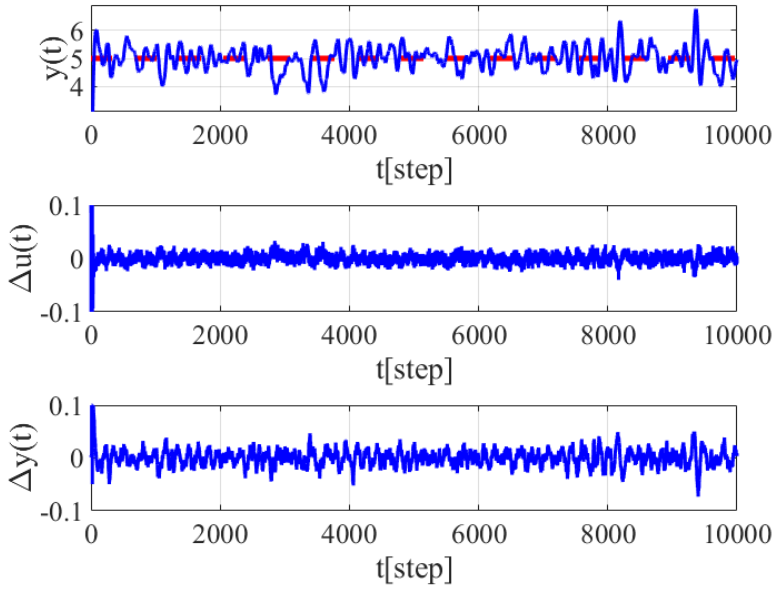

図 4.19: $\Delta K = 0(K = 10), \Delta L = 6(L = 14)$ の制御結果 (クラス K^0, L^+)

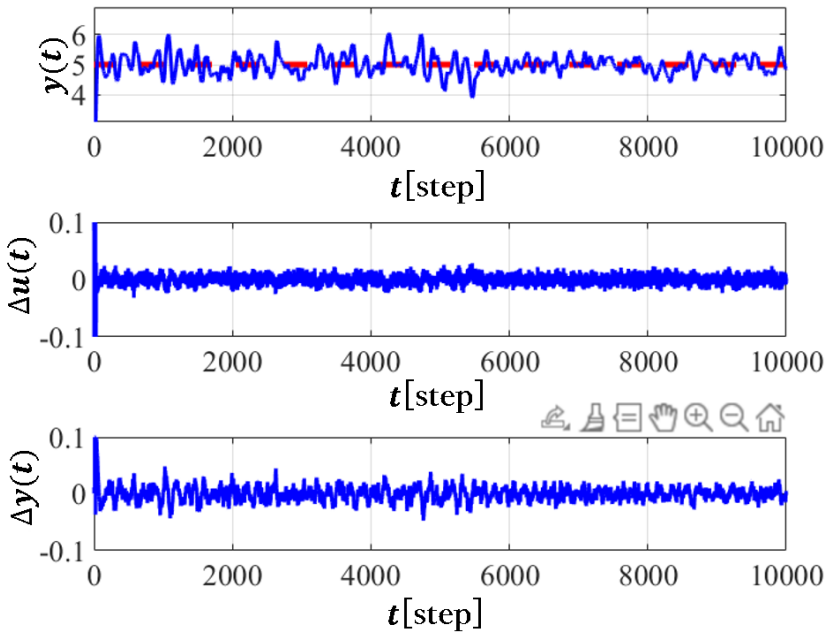

図 4.20: $\Delta K = 7(K = 17), \Delta L = -6(L = 2)$ の制御結果 (クラス K^+, L^-)

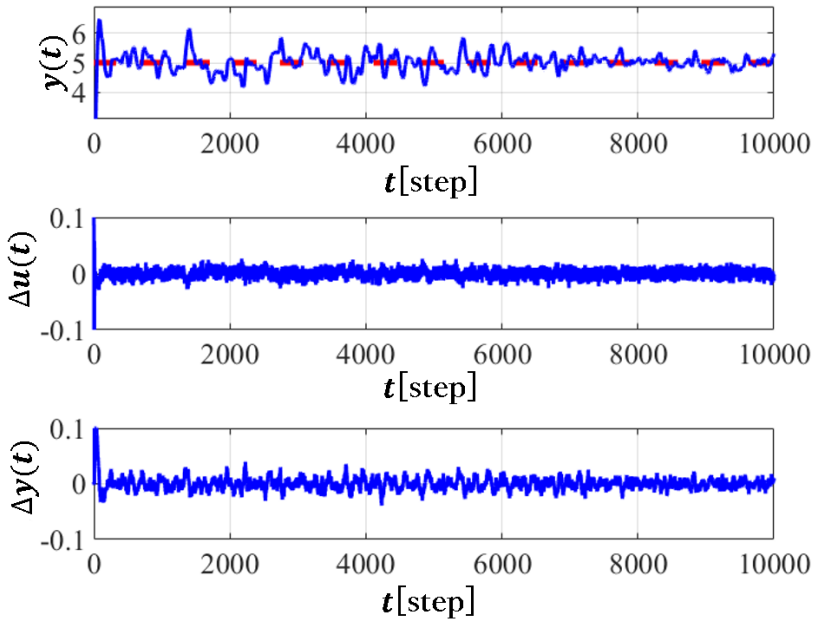

図 4.21: $\Delta K = 7(K = 17), \Delta L = 0(L = 8)$ の制御結果 (クラス K^+, L^0)

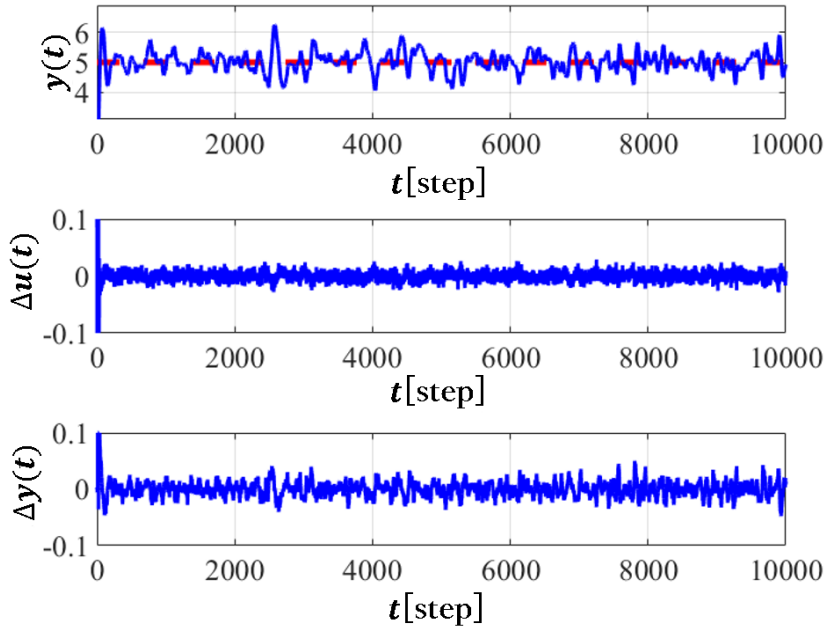

図 4.22: $\Delta K = 7(K = 17), \Delta L = 6(L = 14)$ の制御結果 (クラス K^+, L^+)

K, L が同時に変動する場合の変動分類実験では，複数 step ごとにデータを与えて学習を行う．また，LSTM の出力を O_m^{KL} とし，教師信号は One-hot 表現でそれぞれ与えられる．

KLが同時に変動する場合の変動分類実験における一段階目の学習用データ各100サンプルずつの分類結果を表4.21にまとめる．行は実際に分類されたデータのクラスを表しており，列は学習に用いたデータのサンプル数を示している．

表 4.21: KL 変動分類実験における一段階目の学習用データの分類結果

クラス	K^-, L^-	K^-, L^0	K^-, L^+	K^+, L^+	K^0, L^0	K^0, L^+	K^0, L^-	K^+, L^0	K^+, L^-
K^-, L^-	**100**	0	0	0	0	0	0	0	0
K^-, L^0	0	**100**	0	0	0	0	0	0	0
K^-, L^+	0	0	**100**	0	0	0	0	0	0
K^+, L^+	0	0	0	**100**	0	0	0	0	0
K^0, L^0	0	0	0	0	**100**	0	0	0	0
K^0, L^+	0	0	0	0	0	**100**	0	0	0
K^0, L^-	0	0	0	0	0	0	**100**	0	0
K^+, L^0	0	0	0	0	0	0	0	**100**	0
K^+, L^-	0	0	0	0	0	0	0	0	**100**

表4.21から，変動の特徴に関係したクラスに100%の精度で分類されており，正しく学習が完了していると考えられる．

次に学習を行ったLSTMを用いて未知の検証用データの分類を行う．

KL が同時に変動する場合の変動分類実験における検証用データ各 100 サンプルずつの分類結果を表 4.22 にまとめる．

表 4.22: KL 変動分類実験における一段階目の検証用データの分類結果

クラス	K^-, L^-	K^-, L^0	K^-, L^+	K^+, L^+	K^0, L^0	K^0, L^+	K^0, L^-	K^+, L^0	K^+, L^-
K^-, L^-	**25**	24	26	6	7	5	0	0	0
K^-, L^0	28	**25**	21	0	1	2	0	0	0
K^-, L^+	26	18	**26**	5	3	6	0	0	0
K^+, L^+	8	9	5	**20**	14	18	1	14	6
K^0, L^0	3	6	6	17	**25**	19	1	19	3
K^0, L^+	10	18	16	7	4	**5**	0	5	0
K^0, L^-	0	0	0	14	13	5	**55**	7	3
K^+, L^0	0	0	0	15	13	26	4	**49**	11
K^+, L^-	0	0	0	17	20	14	39	6	**43**

学習用データと比較して変動の特徴に関係したクラスに分類されるデータの割合が少なくなっており，クラスの分類に偏りが生じているものが存在した．この結果から正しいクラスに分類されたデータ数に対して他のクラスに誤分類される比率が大きくなっているデータは似た特徴を持ち分類が困難になっていると考えられる．そのため，誤分類の割合が多いクラスのデータのみを用いて二段階目の学習を行うことが必要であると考え，似た特徴を持つと考えられるデータごとにグループ分けを行い，各グループのデータのみを用いて二段階目の学習及び変動分類実験を行う．

表 4.23 に一段階目の検証用データの分類結果をもとにグループ分けを行った結果をまとめる．表 4.23 をもとに各グループの検証用データを用いて第二段階の学習・分類実験を行う．

表 4.23: 一段階目の分類結果をもとにしたグループ分け

	クラス		
グループ 1	K^-, L^-	K^-, L^0	K^-, L^+
グループ 2	K^0, L^-	K^0, L^0	K^0, L^+
グループ 3	K^+, L^-	K^+, L^0	K^+, L^+

KL 変動分類実験における二段階目の学習用データ各 100 サンプルずつの分類結果を表 4.24-4.26 にまとめる.

表 4.24: KL 変動分類実験における二段階目の学習用データの分類結果 (グループ 1)

クラス	K^-, L^-	K^-, L^0	K^-, L^+
K^-, L^-	**100**	0	0
K^-, L^0	0	**100**	0
K^-, L^+	0	0	**100**

表 4.25: KL 変動分類実験における二段階目の学習用データの分類結果 (グループ 2)

クラス	K^0, L^-	K^0, L^0	K^0, L^+
K^0, L^-	**100**	0	0
K^0, L^0	0	**100**	0
K^0, L^+	0	0	**100**

表 4.26: KT 変動分類実験における二段階目の学習用データの分類結果 (グループ 3)

クラス	K^+, L^-	K^+, L^0	K^+, L^+
K^+, L^-	**100**	0	0
K^+, L^0	0	**100**	0
K^+, L^+	0	0	**100**

表 4.24-4.26 から，変動の特徴に関係したクラスに 100%の精度で分類されており，正しく学習が完了していると考えられる.

次に学習を行った LSTM を用いて各グループに対して二段階目の未知の検証用データの分類を行う.

KL が同時に変動する場合の変動分類実験における二段階目の検証用データ各 100 サンプルずつの分類結果を表 4.27-4.29 にまとめる．

表 4.27: KL 変動分類実験における二段階目の検証用データの分類結果 (グループ 1)

クラス	K^-, L^-	K^-, L^0	K^-, L^+
K^-, L^-	**42**	38	32
K^-, L^0	24	**29**	37
K^-, L^+	34	33	**31**

表 4.28: KL 変動分類実験における二段階目の検証用データの分類結果 (グループ 2)

クラス	K^0, L^-	K^0, L^0	K^0, L^+
K^0, L^-	**85**	22	6
K^0, L^0	11	**46**	15
K^0, L^+	4	32	**79**

表 4.29: KL 変動分類実験における二段階目の検証用データの分類結果 (グループ 3)

クラス	K^+, L^-	K^+, L^0	K^+, L^+
K^+, L^-	**37**	18	38
K^+, L^0	31	**53**	30
K^+, L^+	32	29	**32**

表 4.27 に関して，どのクラスも分類精度が約 40%以下になっており適切に分類することができていない．表 4.28 に関しては，K^0, L^- と K^0, L^+ のクラスの入出力データに関しては約 80%の精度で正しいクラスに分類することができている一方で K^0, L^0 のクラスの入出力データに関しては分類精度が 50%程度と低くなっている．表 4.29 に関しては，表 4.27 と同様に十分な精度で分類することができていない．そのため，KT 変動分類実験と同様に信号処理などの手法を用いることで特徴量を明確化したうえでの学習・分類が必要だと考えられる．

4.7. T, L が同時に変動する場合の変動分類実験

T, L 変動実験のシステム変動は以下の変動式で表わされる．

$$T = \begin{cases} 100 & (t \leqq 5000) \\ 100 + \frac{\Delta T(t-5000)}{2500} & (5000 < t \leqq 7500) \\ 100 + \Delta T & (t > 7500) \end{cases} \tag{4.11}$$

$$L = \begin{cases} L & (t \leqq 5000) \\ L + \frac{\Delta L(t-5000)}{2500} & (5000 < t \leqq 7500) \\ L + \Delta L & (t > 7500) \end{cases} \tag{4.12}$$

ΔT，ΔL はそれぞれシステム変動の大きさを表しており，本実験では ΔT に -60, 60，ΔL に -6，6 のいずれかを代入し，それぞれ学習用データとして用いた．変動後に $T < 100$ のデータをクラス T^-，変動していない $T = 0$ のデータをクラス T^0，$T > 100$ のデータをクラス T^+ とし，変動後に $L < 8$ のデータをクラス L^-，変動していない $L = 8$ のデータをクラス L^0，$L > 8$ のデータをクラス L^+ として同時に変動させ，9 種類のクラスにそれぞれ $(T^*,\ L^*)$，$* \in \{0,\ +,\ -\}$ としてラベルを与える．各クラスに対応するデータを 100 サンプルずつ取得して学習を行う．取得したデータの一例を図 4.23–4.30 に示す．

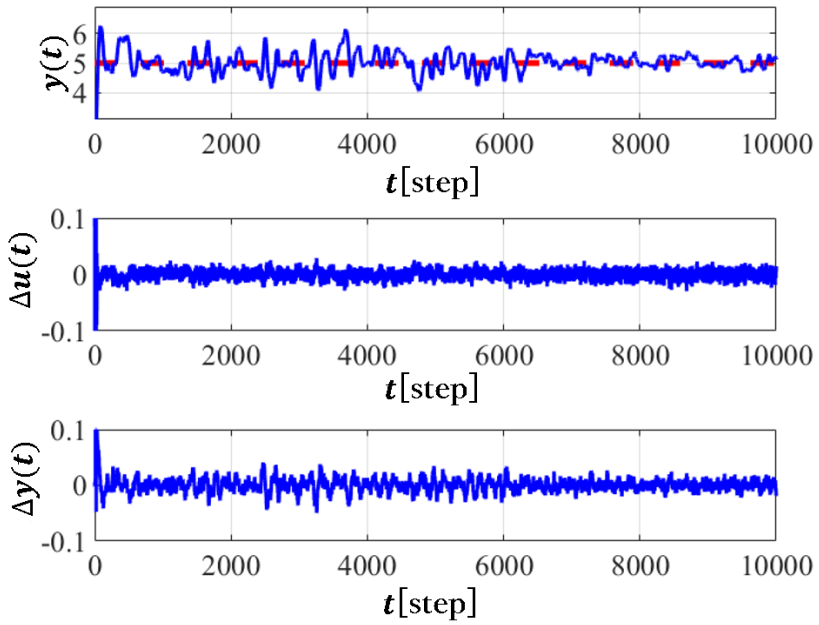

図 4.23: $\Delta T = -60 (T = 40), \Delta L = -6 (L = 2)$ の制御結果 (クラス T^-, L^-)

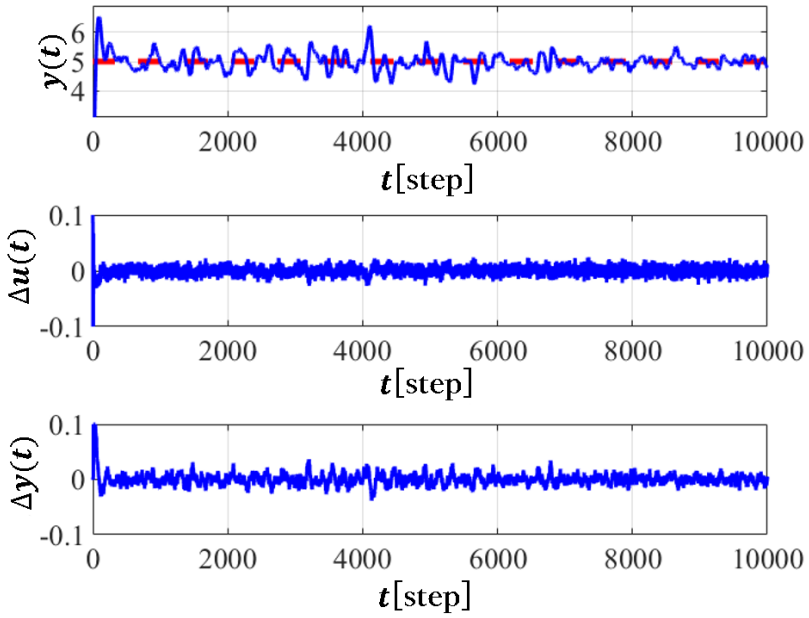

図 4.24: $\Delta T = -60(T = 40), \Delta L = 0(L = 8)$ の制御結果 (クラス T^-, L^0)

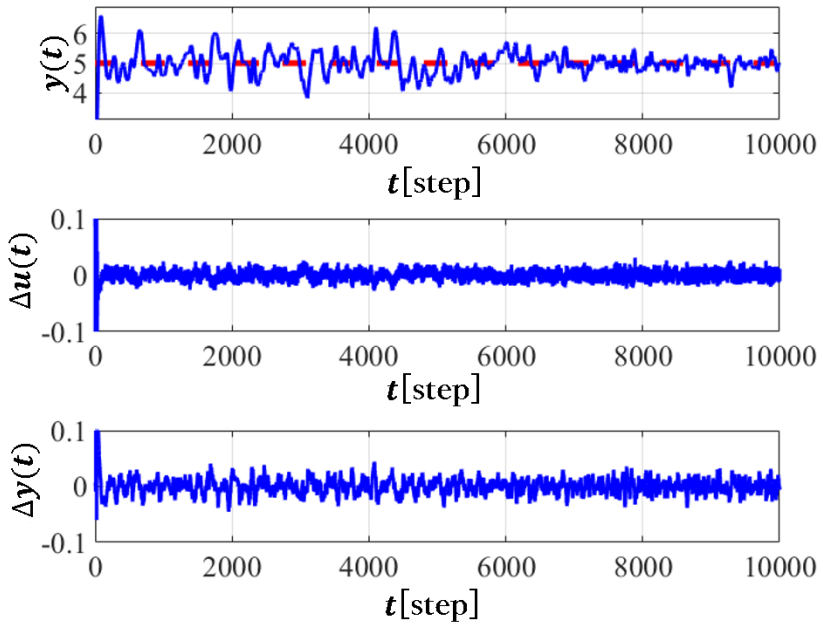

図 4.25: $\Delta T = -60(T = 40), \Delta L = 6(T = 14)$ の制御結果 (クラス T^-, L^+)

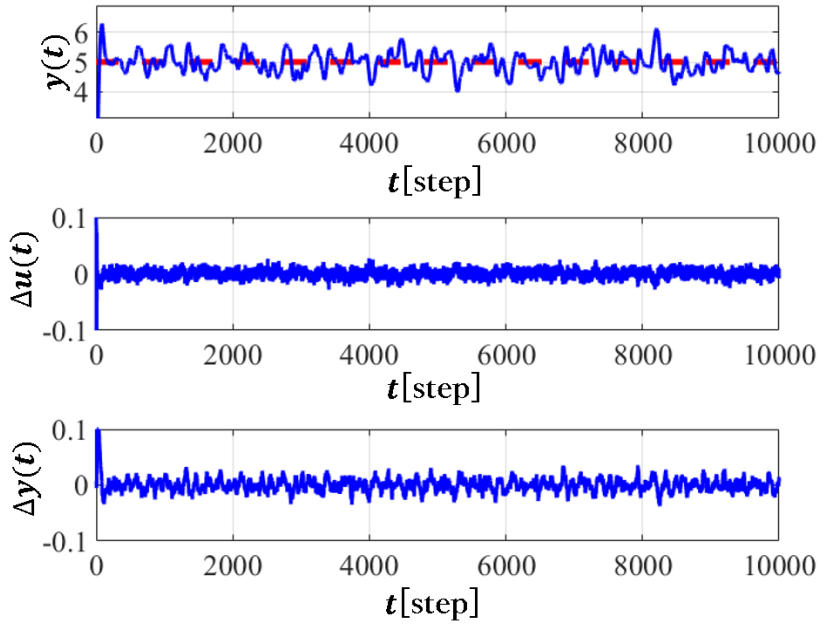

図 4.26: $\Delta T = 0 (T = 100), \Delta L = -6 (L = 2)$ の制御結果 (クラス T^0, L^-)

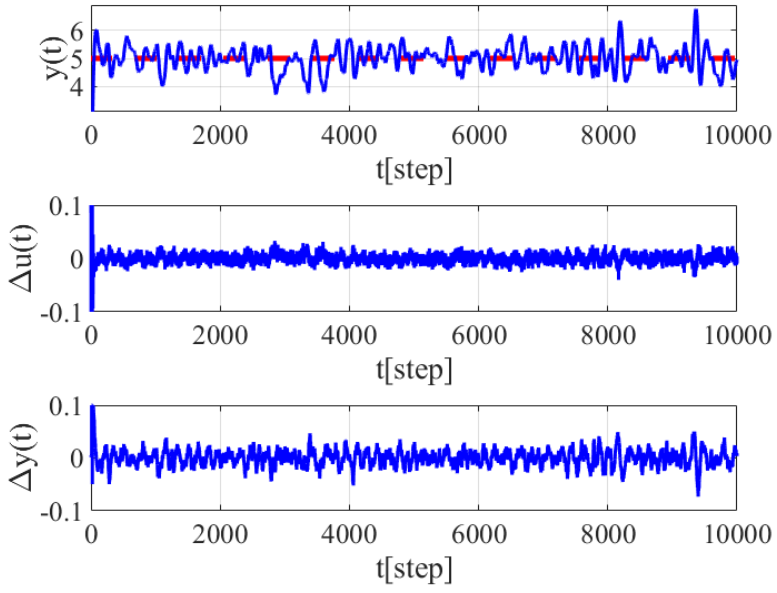

図 4.27: $\Delta T = 0 (T = 100), \Delta L = 6 (L = 14)$ の制御結果 (クラス T^0, L^+)

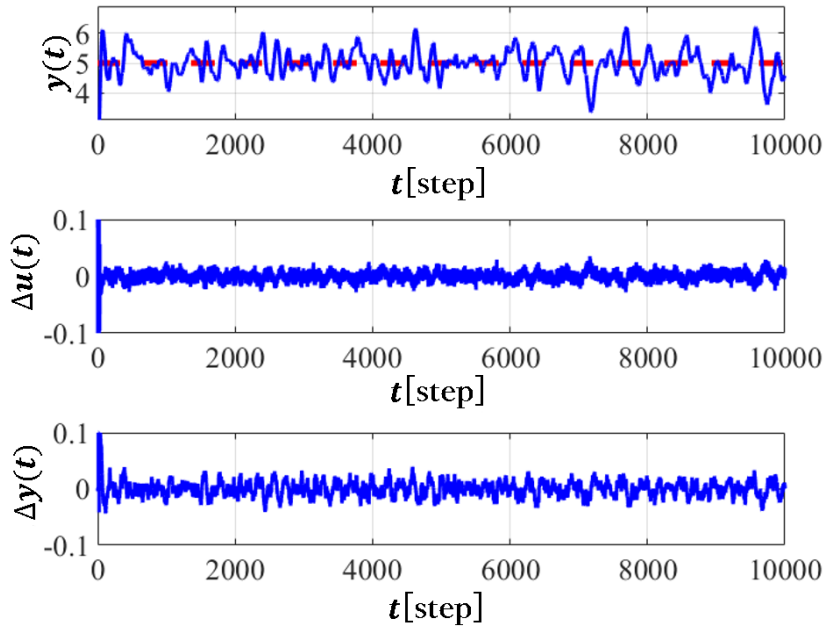

図 4.28: $\Delta T = 60(T = 160), \Delta L = -6(L = 2)$ の制御結果 (クラス T^+, L^-)

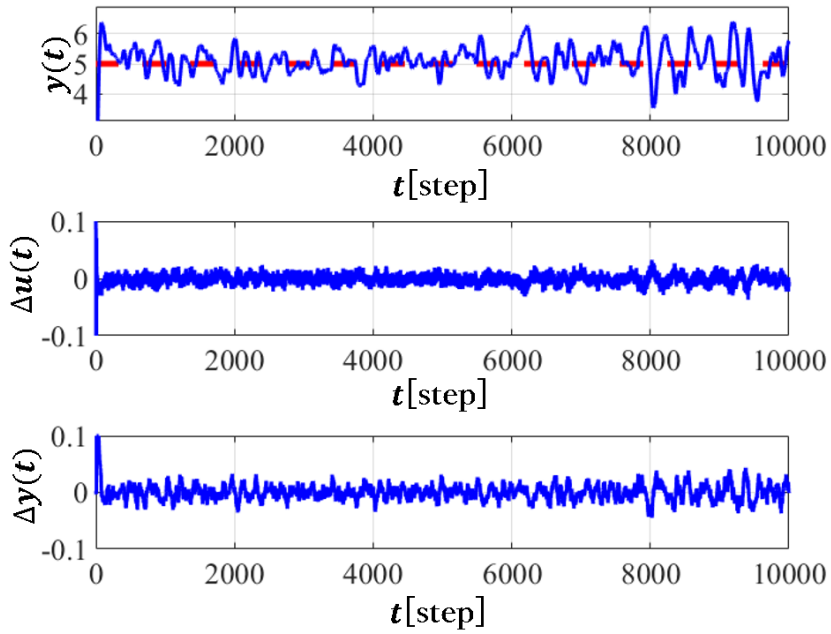

図 4.29: $\Delta T = 60(T = 160), \Delta L = 0(L = 8)$ の制御結果 (クラス T^+, L^0)

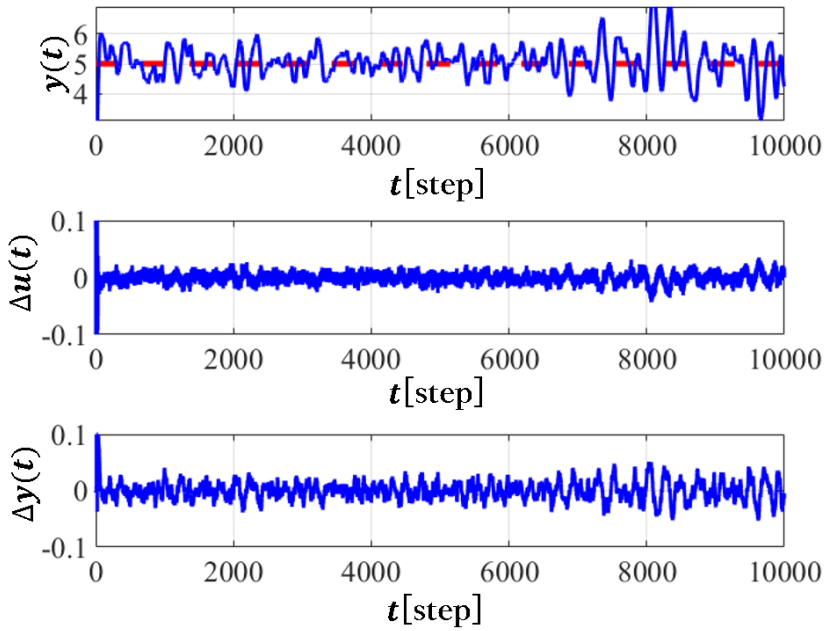

図 4.30: $\Delta T = 60(T = 160), \Delta L = 6(L = 14)$ の制御結果 (クラス T^+, L^+)

T, L が同時に変動する場合の変動分類実験では，複数 step ごとにデータを与えて学習を行う．また，LSTM の出力を O_m^{TL} とし，教師信号は One-hot 表現でそれぞれ与えられる．

TL が同時に変動する場合の変動分類実験における一段階目の学習用データ各 100 サンプルずつの分類結果を表 4.30 にまとめる．行は実際に分類されたデータのクラスを表しており，列は学習に用いたデータのサンプル数を示している．

表 4.30: TL 変動分類実験における一段階目の学習用データの分類結果

クラス	T^-, L^-	T^-, L^0	T^-, L^+	T^0, L^-	T^0, L^0	T^0, L^+	T^+, L^-	T^+, L^0	T^+, L^+
T^-, L^-	**100**	0	0	0	0	0	0	0	0
T^-, L^0	0	**100**	0	0	0	0	0	0	0
T^-, L^+	0	0	**100**	0	0	0	0	0	0
T^+, L^+	0	0	0	**100**	0	0	0	0	0
T^0, L^0	0	0	0	0	**100**	0	0	0	0
T^0, L^+	0	0	0	0	0	**100**	0	0	0
T^0, L^-	0	0	0	0	0	0	**100**	0	0
T^+, L^0	0	0	0	0	0	0	0	**100**	0
T^+, L^-	0	0	0	0	0	0	0	0	**100**

表 4.30 から，変動の特徴に関係したクラスに 100%の精度で分類されており，正しく学習が完了していると考えられる．

次に学習を行った LSTM を用いて未知の検証用データの分類を行う．

TL が同時に変動する場合の変動分類実験における検証用データ各 100 サンプルずつの分類結果を表 4.31 にまとめる．

表 4.31: TL 変動分類実験における一段階目の検証用データの分類結果

クラス	T^-, L^-	T^-, L^0	T^-, L^+	T^0, L^-	T^0, L^0	T^0, L^+	T^+, L^-	T^+, L^0	T^+, L^+
T^-, L^-	**100**	0	0	0	0	1	0	0	0
T^-, L^0	0	**100**	1	0	0	0	0	0	0
T^-, L^+	0	0	**97**	1	1	0	0	0	0
T^+, L^+	0	0	0	**20**	20	54	12	5	0
T^0, L^0	0	0	0	47	**47**	30	8	13	2
T^0, L^+	0	0	2	16	16	**0**	6	9	5
T^0, L^-	0	0	0	8	8	13	**29**	29	19
T^+, L^0	0	0	0	3	3	1	23	**25**	31
T^+, L^-	0	0	0	5	5	1	22	19	**43**

学習用データと比較して T^- が含まれるデータに関しては 90%以上の精度で分類することができているものの，その他のクラスに関しては変動の特徴に関係したクラスに分類されるデータの割合が少なくなっており，クラスの分類に偏りが生じているものが存在した．この結果から正しいクラスに分類されたデータ数に対して他のクラスに誤分類される比率が大きくなっているデータは似た特徴を持ち分類が困難になっていると考えられる．そのため，誤分類の割合が多いクラスのデータのみを用いて二段階目の学習を行うことが必要であると考え，似た特徴を持つと考えられるデータごとにグループ分けを行い，各グループのデータのみを用いて二段階目の学習及び変動分類実験を行う．

表 4.32 に一段階目の検証用データの分類結果をもとにグループ分けを行った結果をまとめる．表 4.32 をもとに各グループの検証用データを用いて第二段階の学習・分類実験を行う．

表 4.32: 一段階目の分類結果をもとにしたグループ分け

	クラス		
グループ 1	T^0, L^-	T^0, L^0	T^0, L^+
グループ 2	T^+, L^-	T^+, L^0	T^+, L^+
グループ 3	T^-, L^-		
グループ 4	T^-, L^0		
グループ 5	T^-, L^+		

TL 変動分類実験における二段階目の学習用データ各 100 サンプルずつの分類結果を表 4.33，4.34 にまとめる．

表 4.33: TL 変動分類実験における二段階目の学習用データの分類結果 (グループ 1)

クラス	T^0, L^-	T^0, L^0	T^0, L^+
T^0, L^-	**100**	0	0
T^0, L^0	0	**100**	0
T^0, L^+	0	0	**100**

表 4.34: TL 変動分類実験における二段階目の学習用データの分類結果 (グループ 2)

クラス	T^+, L^-	T^+, L^0	T^+, L^+
T^+, L^-	**100**	0	0
T^+, L^0	0	**100**	0
T^+, L^+	0	0	**100**

表 4.33，4.34 から，変動の特徴に関係したクラスに 100%の精度で分類されており，正しく学習が完了していると考えられる．

次に学習を行った LSTM を用いて各グループに対して二段階目の未知の検証用データの分類を行う．

KT が同時に変動する場合の変動分類実験における二段階目の検証用データ各 100 サンプルずつの分類結果を表 4.35，4.36 にまとめる．

表 4.35: TL 変動分類実験における二段階目の検証用データの分類結果 (グループ 1)

クラス	T^0, L^-	T^0, L^0	T^0, L^+
T^0, L^-	**83**	22	9
T^0, L^0	13	**46**	14
T^0, L^+	4	32	**77**

表 4.36: TL 変動分類実験における二段階目の検証用データの分類結果 (グループ 2)

クラス	T^+, L^-	T^+, L^0	T^+, L^+
T^+, L^-	**45**	38	26
T^+, L^0	34	**28**	28
T^+, L^+	21	34	**46**

表 4.35 に関して，KL 変動分類実験と同様に L のみが変動した場合のデータに関しては約 80%程度の分類精度で分類することができているが変動していないクラスのデータの分類精度は 50%程度になっている．表 4.36 に関して，50%以下の分類精度でしか分類することができておらず適切に分類することが困難になっていると考えられる．そのため，より精度を高めるためには信号処理などの手法を用いることで特徴量を明確化したうえでの学習・分類が必要であると考えられる．

4.8. 考察

本実験では，単一のシステムパラメータの変動する場合と K, T, L のうち 2 つのパラメータが同時に変動する場合のシステムパラメータ変動に対して LSTM の学習を行った．

単一のシステムパラメータの変動に関して，1step ごとに学習を行った場合に比べて複数 step ごとに学習を行った場合の方がシステム変動の種類に対応したクラスに分類されるデータの割合が多くなった．また，学習にかかる時間に関しても 1step ごとに学習を行った場合に比べて複数 step ごとに学習を行った場合の方が計算回数が削減され効率的に学習を行うことができた．この結果から複数 step ごとに学習を行った場合の方が時系列特性を捉える精度が高いと考えられる．一方でわずかな変動を起こしたデータに対しては，変動を起こしていないクラスに分類される割合が多くなっていた．この結果から，わずかな変動を起こしたデータの変動の特徴量が不明瞭になっていると考えられる．そのため，信号処理などを用いて学習データの前処理を行うことによって特徴量を明確化したうえでの学習・分類を行うことが必要になると考えられる．

2 つのパラメータが同時に変動する場合に関しては，多段階モデルを用いて変動分類を行った．その結果，一段階目には変動の種類を特定することが難しいデータの分類をすることができた．一方で，多段階での変動分類を行っても変動の種類を特定することの難しいデータも存在した．この結果から，二つのシステムパラメータがそれぞれの変動がお互いの変動の特徴を打ち消してしまい特徴が表れにくくなってしまっていると考えられる．そのため，単一のシステムパラメータの変動分類と同様に信号処理などを用いて学習データの前処理を行うことによる特徴量を明確化したうえでの学習・分類やより適切な学習モデルを用いた変動分類が必要になると考えられる．

第 5 章

おわりに

本論文では，多段階モデルを用いてシステム変動の種類を分類する手法を提案した．数値実験において，複数 step ごとに学習用データを与えて学習させることで計算時間を削減することができ概ね良好な分類結果を得ることができた．また，多段階モデルを用いることで 2 つのパラメータが同時に変動する場合の変動分類において，一段階では分類することのできないデータを分類することができた．

ニューラルネットワークは構造のパラメータによって，性能が変化する．ネットワークが小規模の場合，学習データに対する誤差を十分に小さくできず，逆に大規模の場合，必要以上に学習をしてしまう過学習を起こし，検証データに対して，汎化能力が低くなることがあり，対象データに対して必要十分な構造にする必要がある．そのため，文献 [31] のようなリカレントニューラルネットワークの構造最適化を用いることで，分類精度の向上が期待できる．また，現時点でさまざまな構造をもつ LSTM が開発されており，PID 制御の制御パラメータの学習に適した LSTM を検討する必要がある．

今後の課題として，リカレントニューラルネットワークの構造最適化手法を用いた分類精度の向上や，複数の機械学習モデルを用いた多段階での変動分類手法の開発，信号処理などの前処理を行ったうえでの学習・分類手法の開発が考えられる．

謝辞

本論文作成の全過程を通じて終始理解ある御教授，御指導，御鞭撻を賜りました，西崎一郎教授，林田智弘准教授，関崎真也助教に厚く御礼申し上げます.

また，研究生活を通じて，親切な御助力を頂きました社会情報学研究室の皆様，客観的な視点から研究に対するご助言をいただきました，副指導教員の山本透教授にも深く御礼申し上げます.

参考文献

[1] 大松, 山本, “セルフチューニングコントロール ”, コロナ社 , 1996.

[2] 山本, 兼田, “一般化最小分散制御則に基づくセルフチューニング PID 制御器の一設計”, システム制御情報学会論文誌, **11**, pp. 1–9, 1998.

[3] 山本, “モデリング性能評価に基づくパフォーマンス・アダプティブ PID 制御系の設計”, *The Institute of Electrical Engineers of Japan* C, **127**, pp. 2101–2108, 2007.

[4] 林田, 山本, 木下, 西崎, 関崎, 平塚, “リカレントニューラルネットワークを用いたシステム変動検出手法の提案”, 電気学会論文誌 C, **137**, pp. 242–248, 2017.

[5] 水口, 林田, 山本, 木下, 西崎, 関崎, “ リカレントニューラルネットワークを用いた制御系のシステム変動分類 ”, 電気学会研究会資料・システム制御合同研究会「制御工学と機械学習の最新動向」, ST–18–151, CT–18–116, pp. 1–5, 2018.

[6] S. Hochreiter, and J. Schmidhuber. “Longshort-term memory.”, *Neural Computation*, **9**, pp. 1735–1780, 1997.

[7] B.Q. Huang, T. Rashid. and M.-T. Kechadi, “Multi-context recurrent neural network for time series applications”, *World Academy of Science, Engineering and Technology,* **1**, pp. 448–457, 2007.

[8] 森 仁紀, 林田 智弘, 西崎 一郎, 関崎 真也, ”LSTM を用いたシステムの変動検出および分類手法の改良,” 2022 年電気学会 電子・情報・システム部門大会, TC1-6, pp. 26-31, 2022.

[9] A. Rajavelu, M.T. Musavi, and M.V. Shirvaikar, “A neural network approach to character recognition”, *Neural Networks*, **2**, pp.387–393, 1989.

[10] A. Mohamed, G.E. Dahl, and G.E. Hinton, “Deep belief network for phone recognition”, In *NIPS Workshop on Deep Learning for speech Recognition and Related Applications*, 2009.

[11] T. Chenoweth and Z. Obradović, “A multi-component nonlinear prediction

system for the S&P 500 Index", **10**, *Neurocomputing*, pp.275–290, 1996.

[12] M. Mohandes, A. Balghonaim, M. Kassas, S. Rehman, and T.O. Halawani, "Use of Function for Estimating Monthly Solar Radination", **68**, *Solar Energy*, pp.161–168, 2000.

[13] K.J. Åström and T. Hägglund , "The future of PID control", *Control Engineering Practice*, **9**, pp.1163–1175, 2001.

[14] J.G. Ziegler and N.B. Nichols , "Optimum setting for automatic controllers", ASME Trans, **64**, pp.759–768, 1942.

[15] 佐伯 , "制御工学-古典制御からロバスト制御へ-", 朝倉書店, 2013.

[16] 木下, 井上, 大西, 山本, "FRIT を用いたパフォーマンス駆動型制御系の設計とその応用", システム制御情報学会論文誌, **29**, pp.202–209, 2016.

[17] T.J. Harris, "Assessment of closed loop performance", *Canadian Journal of Chemical Engineering*, **67,** pp.856–861, 1989.

[18] 相馬, 金子, 藤井, "一回の実験データに基づく制御器パラメータチューニングの新しいアプローチ — Fictious Reference Iterative Tuning の提案", システム制御情報学会論文誌, **17**, pp.528–536, 2004.

[19] 高尾, 大西, 山本, 雛元, "制御性能評価に基づくパフォーマンス・アダプティブ PID コントローラの設計", 計測自動制御学会論文集, **43**, pp.110–117, 2007.

[20] L. Desborough and T.J. Harris, "Performance assessment measures for univariate feedback control", *The Canadian Journal of Chemical Engineering*, **70**, pp.1186–1197, 1983.

[21] W.S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity", *Bulletin of Mathematical Biophysics*, **5**, pp.115–133, 1943.

[22] M. I. Jordan, "Attractor dynamics and parallelism in a connectionist sequential machine", in *Proceedings of the 8th Annual Conference of the Cognitive Science Society*, Englewood Cliffs, NJ: Erlbaum, pp.531–546, Reprinted in IEEE Tutorials Series, New York: IEEE Publishing Services, 1990, 1986.

[23] J.L. Elman, "Finding structure in time", *Cognitive Science*, **14**, pp.179–211, 1999.

[24] B. de Vries and J.C. Principe, “A theory for neural networks with time delays”, in NIPS-3: *Proceedings of the 1990 conference on Advances in neural information processing systems*, **3**, pp.162–168, San Francisco, CA, USA, 1990. Morgan Kaufmann Publishers Inc.

[25] D.V. Prokhorov, E.W. Saad, and D.C. Wunsch, “Comparative study of stock trend prediction using time delay, recurrent and probabilistic neural networks”, *IEEE Transactions on Neural Networks*, **9**, pp.1456–1470, 1998.

[26] B.Q. Huang, T. Rashid, and M.T. Kechadi, “Multi-context recurrent neural network for time series application”, *International Journal of Computational Intelligence*, **3**, pp.45–54, 2006.

[27] H. Katagiri, I. Nishizaki, T. Hayashida, and T. Kadoma, “Multiobjective evolutionary optimization of training and topology of recurrent neural networks for time-series prediction”, *The Computer Journal*, **55**, pp.325–336, 2012.

[28] T. Kohonen, “Self-organized formation of topologically correct feature maps”, *Biological Cybernetics*, **43**, pp.59–69, 1982.

[29] D.E. Rumelhat, G.E. Hinton, and R.J. Williams, “Learning international representations by error propagation”, *Parallel distributed processing: explorations in the microstructure of cognition*, **1**, pp.318–362, 1986.

[30] P.J. Werbos, “Backpropagation through time: What it does and how to do it”, *Proceedings of the IEEE*, **78**, pp.1550–1560, 1990.

[31] 林田, 西崎, 末宗, “タブー探索を用いたリカレントニューラルネットワークの構造最適化”, 日本知能情報ファジィ学会誌, **27**, pp.638–649, 2015.