令和元年度　卒業論文

Deep Learning を用いた天気予測

Weather forecast using Deep Learning

長野工業高等専門学校
電子情報工学科 5 年
西村研究室

清水 翔仁

指導教員: 西村 治

令和 2 年 2 月 17 日

目次

1章　はじめに

現在の情報技術は，第四次産業革命とも呼ばれるほど大きな変化をもたらすと捉えられている．特に「人工知能（AI)」は，火薬と核兵器に次ぐ人類最大の発明とも称され，私達の将来を左右する力を持つと考えられる．また 2000 年代に「Deep Learning」の手法が開発されて以来，第三次 AI ブームが起こり，現在も進行中である．

AI 自らが知識を獲得することを「機械学習」という．Deep Learning は，機械学習の一種であるが，ニューラルネットワーク（NN）と呼ばれる人間の脳のメカニズム同様，大量の情報を多層的に処理する手法を用いる点が，従来の機械学習と異なっている．その多層的な情報処理の能力は，注目すべき点である特徴量を人間に支持されなくとも自ら見つけ出すことができる．したがって Deep Learning は，人間が従来認識していなかったような注目点や重要度を提示して「これが知である」という定義を行うことにより，人間の知能を超える可能性を持っている．[1]

Deep Learning 技術は画像認識の分野をはじめとして，音声認識，自然言語処理の分野で大きな成果をあげているが，近年では気象予測の分野でもその適用例が報告されている．文献 [2] によると，現在気象庁が行っている天気予報では，「数値予報ガイダンス」と呼ばれる手法が使われている．数値予報ガイダンスとは，図 1.1 のように，まず気象観測データから物理学の方程式を用いて風や気温などの時間変化を計算する．この計算によって得られた予測結果を数値予報モデルという．その後，予測結果を機械学習を用いて翻訳・修正し，天気予報の精度向上を図る方法である．

この方法では，機械学習の入力となるのは，数値予報モデルから得られた未来の観測データである．また，天気の予測自体には機械学習を使用していない．そこで本研究では，過去の気象観測データを入力データとし，機械学習を用いて翌日の降水量の有無を予測する．

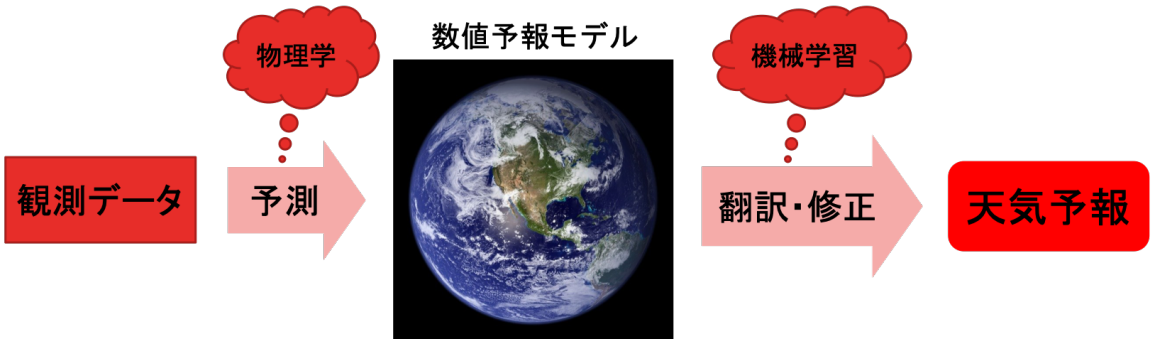

図 1.1: 数値予報ガイダンスの模式図

2章　原理

本章では，本研究で用いた手法の原理について説明する．

2.1　RNN(Recurrent Neural Network)

RNN とは，Deep Learning の手法の一種であり，時系列データの分析に特化している．図 2.1 に，RNN の模式図を示す．ここで，x_t を時刻 t における RNN の入力，h_t を時刻 t における RNN の出力とする．

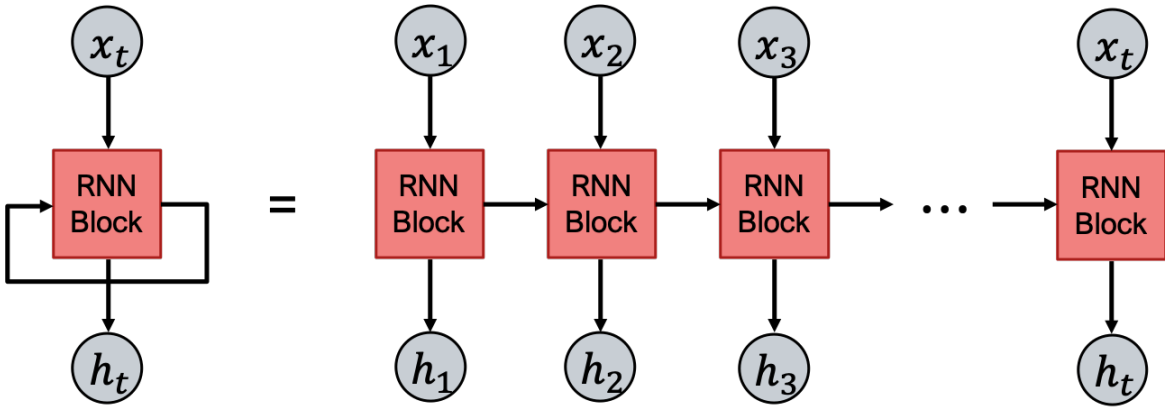

図 2.1: 展開された RNN

図 2.1 のように，RNN は内部にループ構造を持っている．これにより，前のステップでの分析結果を記憶し，データの時系列を理解することができる．しかし，RNN には長期依存性問題という欠点がある．長期依存性問題とは，記憶するステップ数が膨大になると計算が爆発するという問題である．そのため，現在単純な RNN はあまり使用されていない．

2.2　LSTM(Long Short Term Memory)

LSTM とは，RNN の長期依存性問題を解決した手法である．図 2.2 に RNN の内部を，図 2.3 に LSTM の内部を示す．このとき，σ は 0〜1 を出力する関数，tanh は-1〜1 を出力する関数とする．

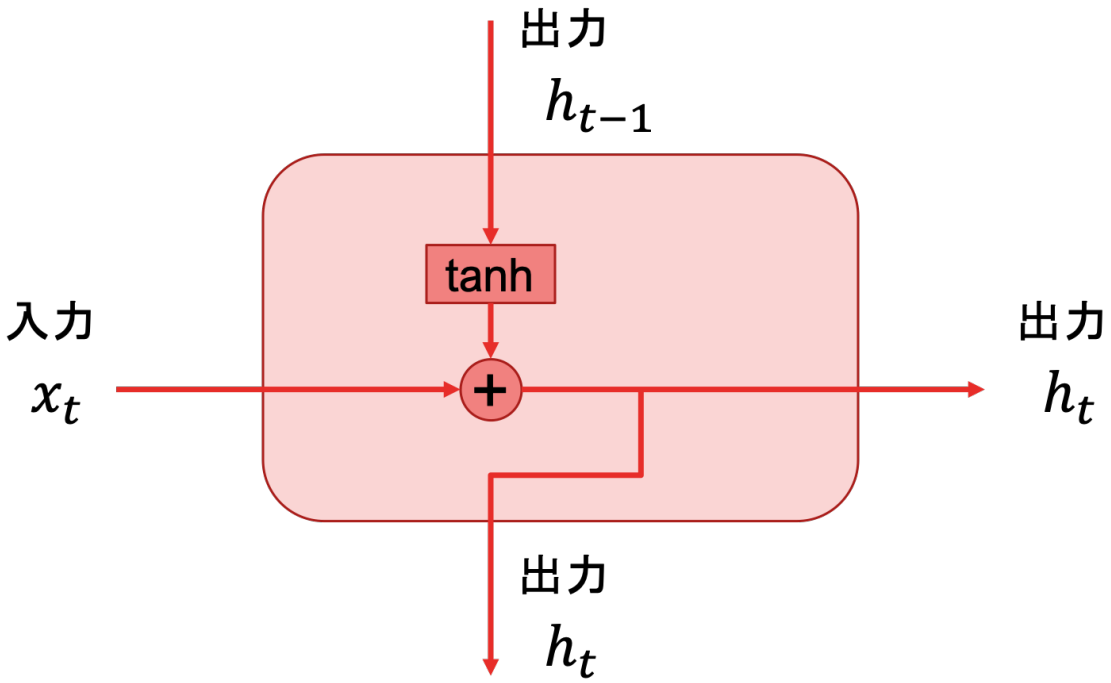

図 2.2: RNN の内部

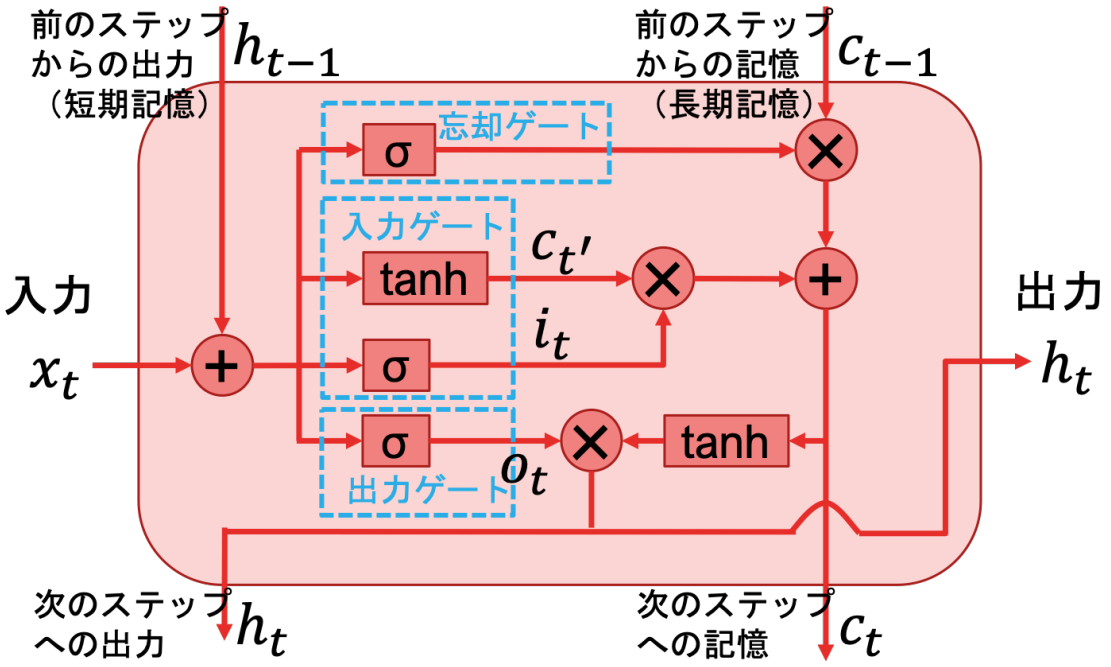

図 2.3: LSTM の内部

図 2.2，図 2.3 より，RNN は単一の tanh 関数という非常に単純な構造に比べ，LSTM は 4 つの関数を含む複雑な構造をしている．

ここで，図 2.3 の LSTM 内にデータが流れる手順について説明する．

1. 前のステップからの出力 h_{t-1} と入力 x_t が合流する．合流した信号はコピーされて 4 つのラインに分岐する．
2. 一番上のラインの忘却ゲートでは，前のステップからの記憶一つ一つに対して，σ 関数からの 0〜1 の値によって情報の取捨選択を行う．このとき 1 は情報を全て残し，0 は全て捨てる．これにより，不要と思われる情報を捨てることで計算の爆発を防ぐ．
3. 入力ゲートにおいて，前のステップからの出力 h_{t-1} と入力 x_t の合算を長期保存用に変換した上で，どの信号をどのくらいの重みで記憶に保存するか制御する．これは 2 つの手順で処理する．
   (a) tanh 関数を用いて，入ってきた情報の情報量を削減し，必要な情報だけに変換された $c_{t'}$ が出力する．
   (b) σ 関数の出力 i_t によって，h_{t-1} を考慮して入力 x_t の重みを調整する．
4. 出力ゲートにおいて，上記の処理で取捨選択された長期記憶 c_t の中で，短期記憶 h_t に関する部分のみを出力する．これも 2 つの手順で処理する．
   (a) 前のステップからの記憶 c_{t-1} と，入力 x_t を変換した短期記憶 $c_{t'}$ を合算し，長期記憶 c_t として出力する．これは，それぞれ既に忘却ゲートおよび入力ゲートで取捨選択が行われている．
   (b) tanh 関数に c_t を入力したものに対し，σ 関数からの 0〜1 の値 o_t によって情報の取捨選択を行う．

3 章　予備知識

本章では，本研究を説明する上で必要な用語について述べる．

3.1　モデル

モデルとは，コンピュータが分かる形の入力値を受け取り，何かしらの評価・判定をして出力値を出すものである．機械学習や AI において中心的な役割を担う頭脳を意味する．

3.2　エポック数

エポック数とは，モデルが学習データセットに対して学習した回数である．エポック数が多すぎると「過学習」を起こし，正確な結果が得られない．逆に少なすぎると，モデルの学習が足りないため精度が伸びない．

3.3　過学習

過学習とは，トレーニングデータセットではうまく機能するモデルが，未知のデータ（テストデータセット）ではうまく汎化されないという問題のことである．その原因は，データに対してモデルが複雑過ぎることや，トレーニングデータセットが足りないことによる学習不足だと考えられる．経験則として，学習中にテストデータの損失が一旦下がったあとに増加したら，それは過学習を起こしている兆候である．[4]

3.4　隠れ層

隠れ層とは，ニューラルネットワークにおける入力層と出力層以外の層のことである．一般に，隠れ層の数や，一つの層あたりのノード数が多ければ多いほど精度が高くなる．しかし多すぎるとモデルが複雑になってしまい，過学習を引き起こす．

3.5 バッチサイズ

学習データセットをいくつかに小分けしたかたまりのことをバッチといい，その大きさのことをバッチサイズという．

3.6 正規化

機械学習では，学習データのそれぞれの値によって大きく桁が異なると，良い性能が得られないという性質がある．そこでデータの値をすべて 0〜1 の間に収めることを正規化という．サンプル $x^{(i)}$ の正規化した値 $x^{(i)}_{norm}$ を求める式を式 (3.1) に示す．式 (3.1) において，$x^{(i)}$ は特定のサンプルであり，$x^{(i)}_{min}$ は特徴量の列における最小値，$x^{(i)}_{\max}$ は最大値を表す．[5]

$$x^{(i)}_{norm} = \frac{x^{(i)} - x_{\min}}{x_{\max} - x_{\min}} \tag{3.1}$$

4 章　実装方法

本章では，開発環境や精度向上のために行ったことについて説明する．

4.1　開発環境

本研究の開発環境を表 4.1 に示す．

表 4.1: 開発環境

OS	macOS Catalina 10.15.2
CPU	Intel(R) Core i7-5650U CPU@2.20GHz
プログラミング言語	Python 3.7.4
ニューラルネットワークライブラリ	keras 2.3.1
デバッグツール	TensorBoard 2.1.0

4.2　学習データセット

今回の学習データセットは，気象庁ホームページ [3] にて公開されている気象観測データをダウンロードして使用する．ダウンロードするにあたって指定した条件を以下に示す．

- データ形式:CSV
- 期間:1998/1/1 - 2019/12/31
- 地点:長野市
- 特徴量:日平均現地気圧，日平均気温，日最低気温，日最高気温，降水量の日合計，日照時間，日平均風速，日最大風速，日平均相対湿度，日最小相対湿度，日平均雲量

次に，実際にダウンロードしたデータの例を表 4.2 に示す．表 4.2 より，この状態では気圧の値のみが明らかに桁が大きいため，全体に正規化の処理を施す．

表 4.2 の値を正規化したものを表 4.3 に示す．表 4.3 より，全ての値が 0〜1 の間に収まっており，大小関係も変化していないことがわかる．

表 4.2: 学習データの例

日付	気圧 [hPa]	平均気温 [°C]	最高気温 [°C]	最低気温 [°C]
1998/1/1	965.8	0.4	5.5	-3.9
1998/1/2	968.3	3.2	8.0	0.0
1998/1/3	969.9	2.3	9.8	-3.0
1998/1/4	960.8	3.2	9.7	-1.1

表 4.3: 正規化後の学習データ

日付	気圧	平均気温	最高気温	最低気温
1998/1/1	0.618	0.189	0.218	0.205
1998/1/2	0.670	0.263	0.277	0.303
1998/1/3	0.704	0.239	0.320	0.227
1998/1/4	0.514	0.263	0.318	0.275

5 章　結果

ここでは，隠れ層の数によって予測の精度がどの程度向上するのかを調べるため，隠れ層の数が 1 つの場合，2 つの場合，3 つの場合の 3 パターンについてプログラムを実行する．

まず隠れ層が 1 つの場合についての結果を図 5.1，図 5.2 に示す．図 5.1 より，精度が右肩上がりに上昇し，90% 近くに達していることがわかる．また図 5.2 より，損失が右肩下がりに減少し，値が 0.25 ほどになっていることがわかる．

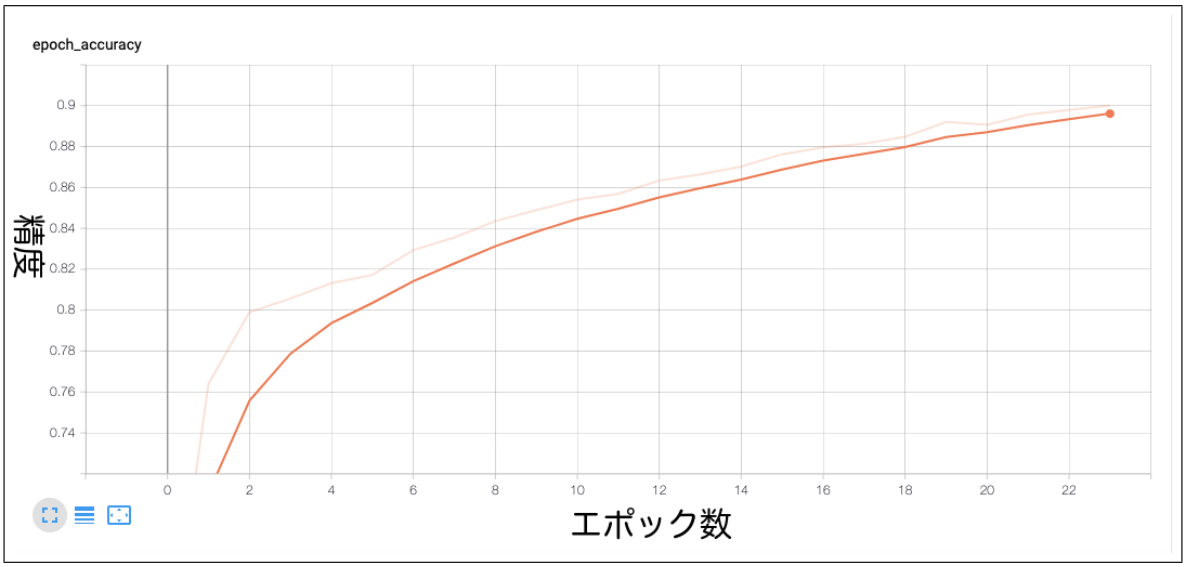

図 5.1: 隠れ層が 1 つの場合の精度

次に隠れ層が 2 つの場合についての結果を図 5.3，図 5.4 に示す．図 5.3 より，精度が 99% に達し図 5.1 よりも高い精度が得られている．また図 5.4 より，損失が 0 に近い値になり図 5.2 よりも減少していることがわかる．このことから，隠れ層の数を増やすことが精度向上に寄与していると考えられる．

隠れ層が 3 つの場合についての結果を図 5.5，図 5.6 に示す．図 5.5 より，エポック数が 7 の時点で精度の上昇が頭打ちになり，エポック数 8 で減少していることがわかる．また図 5.6 より，下がっていた損失がエポック数 7 で上昇していることがわかる．このことから，隠れ層を増やしたことによりニューラルネットワークが複雑になり，過学習が発生していると考えられる．また，以上の結果から隠れ層の数は 2 つが適当であると考えられる．

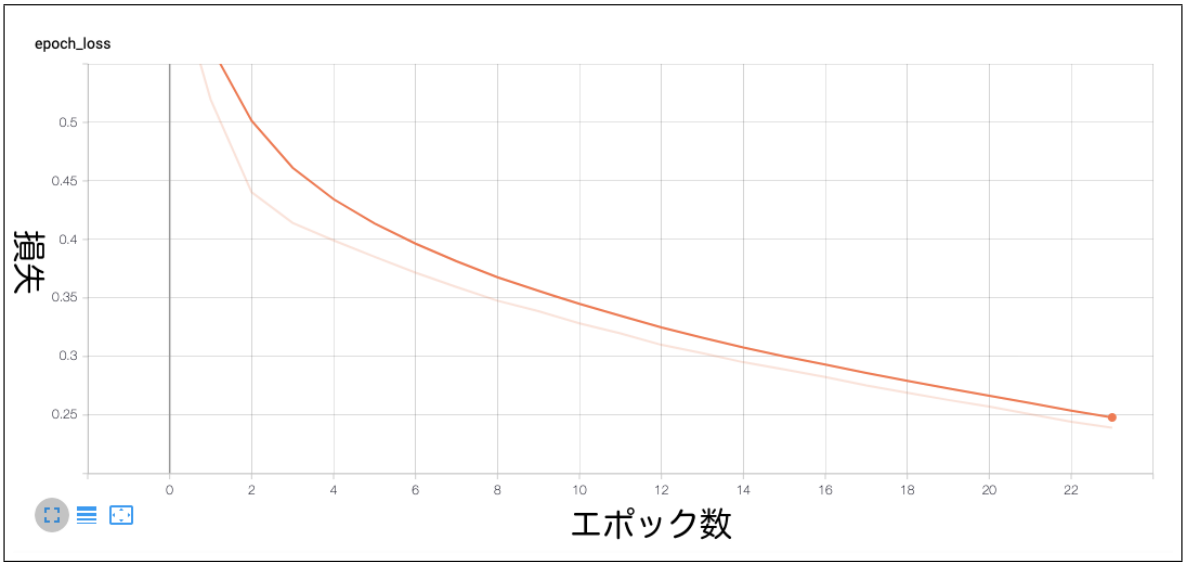

図 5.2: 隠れ層が 1 つの場合の損失

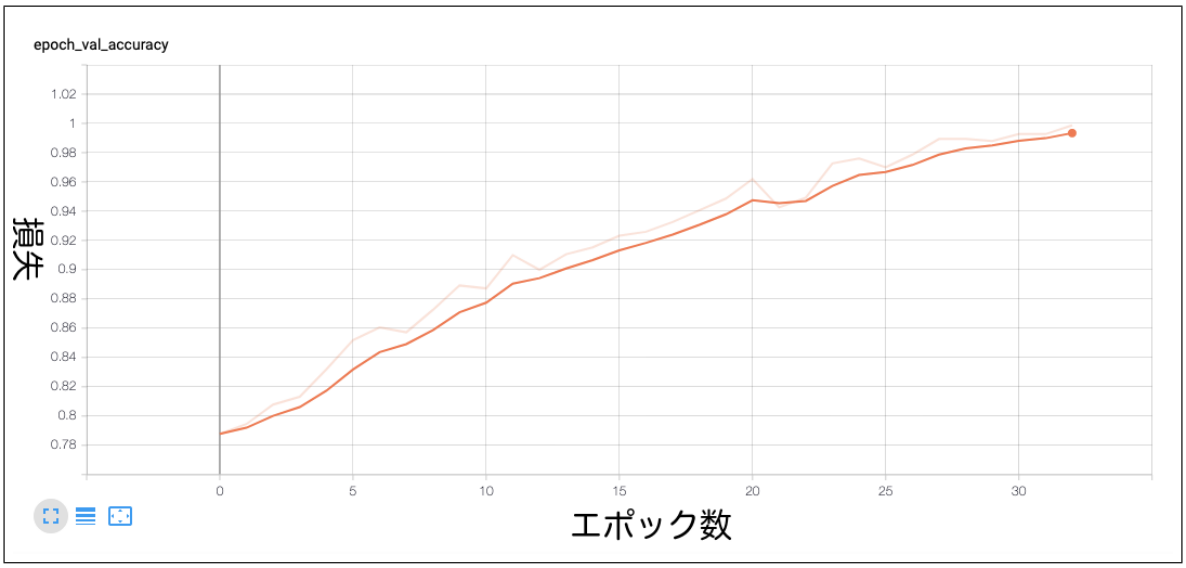

図 5.3: 隠れ層が 2 つの場合の精度

隠れ層の数を 2 つに固定し，プログラムを 10 回実行した際の精度を表 5.1 に示す．表 5.1 より，予測の精度は約 94% となった．よって，かなり高い精度で降水量の有無を予測できていると考えられる．

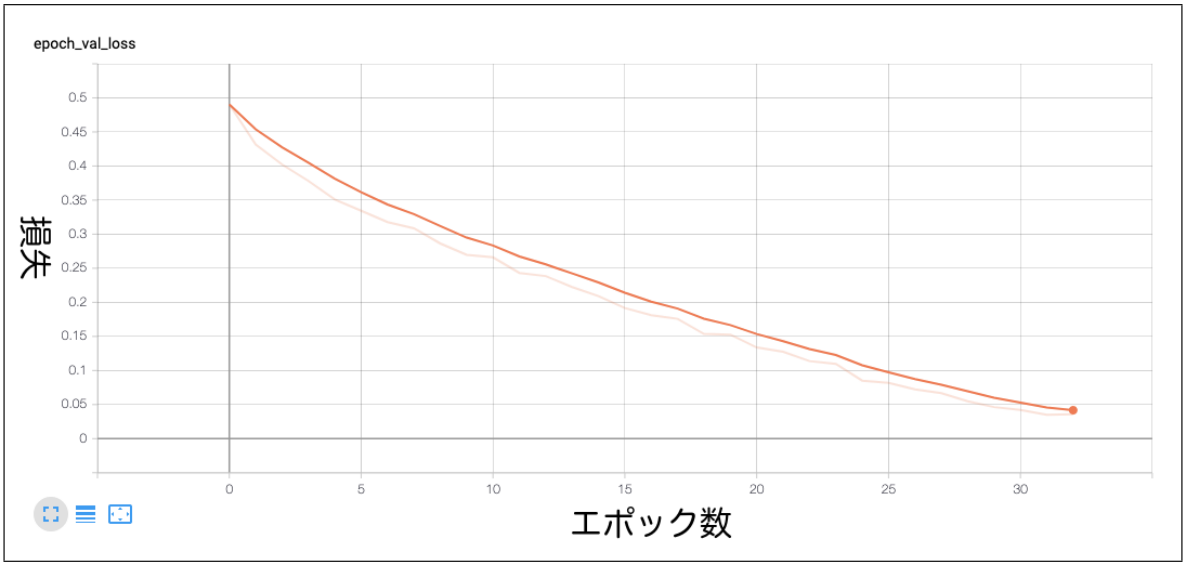

図 5.4: 隠れ層が 2 つの場合の損失

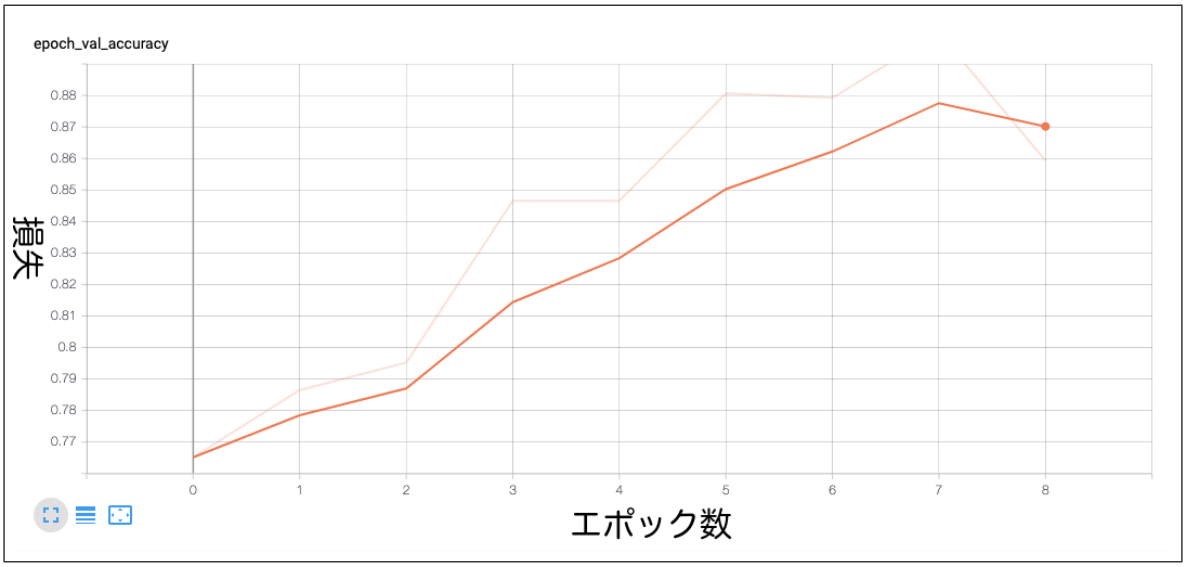

図 5.5: 隠れ層が 3 つの場合の精度

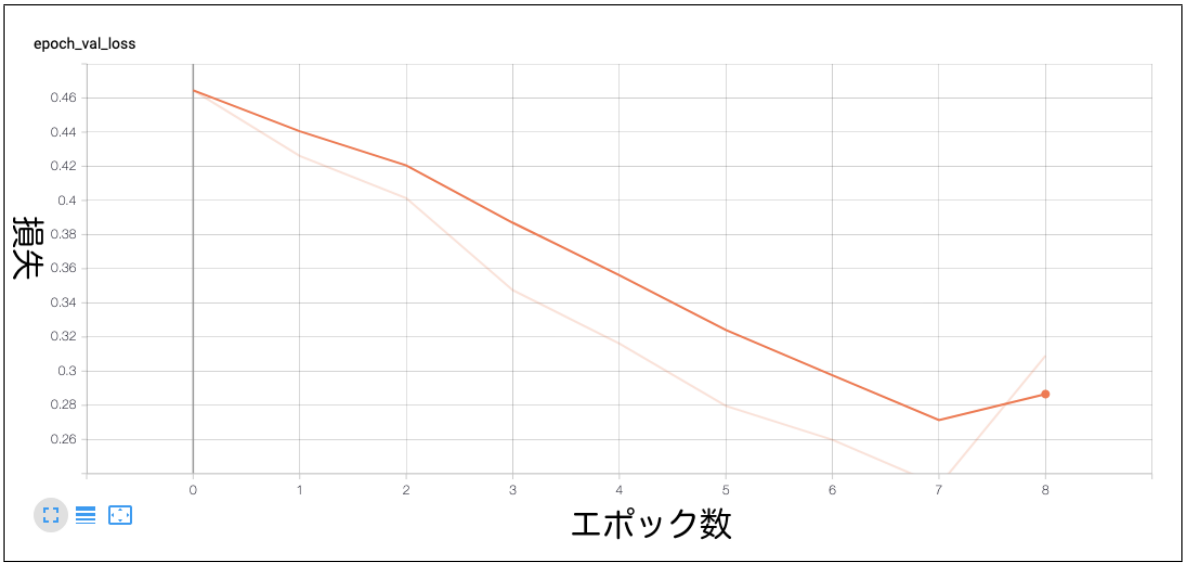

図 5.6: 隠れ層が 3 つの場合の損失

表 5.1: 予測精度

番号	精度
1	0.9248
2	0.9845
3	0.8873
4	0.9028
5	0.9521
6	0.9578
7	0.9560
8	0.9601
9	0.9380
10	0.9091
平均	0.93725

6章　まとめ

本研究の目標である，過去の気象観測データから翌日の降水量の有無を予測することができた．また研究を進める中で，機械学習を行うプログラムの，実装方法を学ぶことができた．

改善点としては，現在日別に出力している予測結果を，より詳細な 1 時間毎の結果に拡大すること．そして降水の有無ではなく，どれくらい雨が降るか，というような降水量の予測を行うことが挙げられる．

謝辞

本研究を進めるに当たり，西村治教授から多大な助言を賜りました．厚く感謝を申し上げます．また一年間同じ研究室で研究を行ってきた青沼葵さん，櫻井優太さん，関谷賢二さん，山本七海さんにも感謝の意を表します．

2020 年 2 月
清水翔仁

参考文献

[1] 鬼頭 葉子: 技術者の倫理, ナカニシヤ出版, 2018.

[2] 気象庁 — 数値予報課報告・別冊第 64 号 (令和 2 年 2 月 16 日現在): https://www.jma.go.jp/jma/kishou/books/nwpreport/64/chapter1.pdf

[3] 気象庁 — 過去の気象データ・ダウンロード (令和 2 年 2 月 16 日現在): https://www.data.jma.go.jp/gmd/risk/obsdl/index.php

[4] Antonio Gulli・Sujit Pal(著)，大串正矢・久保隆宏・中山光樹 (訳): 直感 Deep Learning, 株式会社オライリー・ジャパン, 2019.

[5] Sebastian Raschka・Vahid Mirjalili(著)，株式会社クイープ (訳): [第 2 版]Python 機械学習プログラミング, 株式会社インプレス, 2018.

[6] TensorBoard(令和 2 年 1 月 20 日現在): https://www.tensorflow.org/tensorboard

付録 A　ソースコード

本研究で実装したプログラムのソースコードをソースコード A.1 に示す.

ソースコード A.1: 翌日の降水量の有無を予測するプログラム

```
1  import pandas as pd
2  from sklearn.model_selection import train_test_split
3  import numpy as np
4  from tensorflow.compat.v1.keras.callbacks import EarlyStopping
5  from tensorflow.compat.v1.keras.layers import Activation, Dense, LSTM
6  from tensorflow.compat.v1.keras.models import Sequential
7  from make_tensorboard import make_tensorboard
8  np.random.seed(1234)
9
10 df_nagano = pd.read_csv('./dataset.csv')
11 # 正規化
12 df_nagano = (df_nagano - df_nagano.min()) / (df_nagano.max() - df_nagano.min())
13 # 正解ラベルを作成
14 yLabel = []
15 for i in range(len(df_nagano)-1):
16     yLabel.append(1 if df_nagano.Prec[i] > 0. else 0)
17 df_label = pd.DataFrame({'label': yLabel})
18 df = pd.concat([df_nagano, df_label], axis=1).dropna()
19 # 切片を追加
20 df['intercept'] = 1
21 # 学習用データとテスト用データに分割
22 xTrain, xTest, y_train, y_test = train_test_split(df.drop("label", axis=1),
23                                                   df.label, test_size=0.2)
24 # 3次元データに変換
25 X_train = xTrain.values[:, np.newaxis, :]
26 X_test = xTest.values[:, np.newaxis, :]
27
28 # ハイパーパラメーター設定
29 HIDDEN_LAYER_SIZE = 12
30 BATCH_SIZE = 32
31 NUM_EPOCHS = 128
32
```

```
33 # モデル定義
34 model = Sequential()
35 model.add(LSTM(HIDDEN_LAYER_SIZE,
36             input_shape=(1, HIDDEN_LAYER_SIZE)))
37 model.add(Dense(1))
38 model.add(Activation("sigmoid"))
39 model.summary()
40
41 # コールバック関数
42 es_cb = EarlyStopping(monitor='val_loss', patience=0, verbose=0, mode='auto')
43 tb_cb = make_tensorboard(set_dir_name='./logs/precaution')
44
45 model.compile(loss="binary_crossentropy",
46                 optimizer="adam", metrics=["accuracy"])
47 history = model.fit(X_train, y_train, batch_size=BATCH_SIZE,
48                         epochs=NUM_EPOCHS,
49                         callbacks=[es_cb, tb_cb],
50                         validation_data=(X_test, y_test))
```