The Journal of the Convergence on Culture Technology (JCCT)
Vol. 9, No. 4, pp.487-494, July 31, 2023. pISSN 2384-0358, eISSN 2384-0366

http://dx.doi.org/10.17703/JCCT.2023.9.4.487

JCCT 2023-7-58

R과 텐서플로우 딥러닝 성능 비교

A Deep Learning Performance Comparison of R and Tensorflow

장성봉*

Sung-Bong Jang*

요 약 본 연구에서는 무료 딥러닝 도구인 R과 텐서플로우에 대한 성능 비교를 수행하였다. 실험에서는 각 도구를 사용하여 6종류의 심층 신경망을 구축하고 10년간의 한국 온도 데이터셋을 사용하여 신경망을 학습시켰다. 구축된 신경망의 입력층 노드 갯수는 10개, 출력층은 5개로 설정 하였으며, 은닉층은 5, 10, 20개로 설정하여 실험을 진행 하였다. 학습 데이터는 2013년 3월 1일부터 2023년 3월 29일까지 서울시 강남구에서 수집된 온도 데이터 3681건을 사용하였다. 성능 비교를 위해, 학습된 신경망을 사용하여, 5일간의 온도를 예측하고 예측된 값과 실제값을 사용하여 평균 제곱근 오차(root mean square error, RMSE)값을 측정하였다. 실험결과, 은닉층이 1개인 경우, R의 학습 오차는 0.04731176이었으며, 텐서플로우는 0.06677193으로 측정되었으며, 은닉층이 2개인 경우에는 R이 0.04782134, 텐서플로우는 0.05799060로 측정되었다. 전체적으로 R이 더 우수한 성능을 보였다. 우리는 기계학습을 처음 접하는 사용자들에게 두 도구에 대한 정량적 성능 정보를 제공함으로써, 도구 선택에서 발생하는 어려움을 해소하고자 하였다.

주요어 : 딥러닝 도구, 성능 비교, 기계 학습, 심층 신경망

Abstract In this study, performance comparison was performed on R and TensorFlow, which are free deep learning tools. In the experiment, six types of deep neural networks were built using each tool, and the neural networks were trained using the 10-year Korean temperature dataset. The number of nodes in the input layer of the constructed neural network was set to 10, the number of output layers was set to 5, and the hidden layer was set to 5, 10, and 20 to conduct experiments. The dataset includes 3600 temperature data collected from Gangnam-gu, Seoul from March 1, 2013 to March 29, 2023. For performance comparison, the future temperature was predicted for 5 days using the trained neural network, and the root mean square error (RMSE) value was measured using the predicted value and the actual value. Experiment results shows that when there was one hidden layer, the learning error of R was 0.04731176, and TensorFlow was measured at 0.06677193, and when there were two hidden layers, R was measured at 0.04782134 and TensorFlow was measured at 0.05799060. Overall, R was measured to have better performance. We tried to solve the difficulties in tool selection by providing quantitative performance information on the two tools to users who are new to machine learning.

Key words : Deep Learning Tool, Performance Comparison, Machine Learning, Deep Neural Networks

*정회원, 금오공과대학교 산학협력단 교수 (단독저자)
접수일: 2023년 5월 8일, 수정완료일: 2023년 5월 21일
게재확정일: 2023년 7월 1일

Received: May 8, 2023 / Revised: May 21, 2023
Accepted: July 1, 2023
*Corresponding Author: sungbong.jang@kumoh.ac.kr
Dept. of Industry-Academy, Kumoh Institute of Technnology, Korea

Ⅰ. 서 론

딥러닝 기술은 1950년대부터 발전이 이루어졌으며, 최근 chatGPT와 같은 생성 AI기술의 등장으로 더욱더 많은 관심을 받고 있다[1]. 딥러닝에서는 인공 신경망을 구축하고 대량의 데이터를 사용하여 구축된 신경망을 학습 시킨후, 학습된 신경망을 이용하여 데이터 예측, 이미지 분류, 텍스트 생성과 같은 다양한 작업에 활용된다[2]. 딥러닝 기술을 적용하기 위해서는 대량의 학습 데이터와 적절한 소프트웨어 도구가 필요하다. 현재까지 다양한 딥러닝 도구가 개발되어 제공되고 있으나, 초보자의 경우, 사용하기가 어렵고 비용이 많이 들기 때문에, 쉽게 접근할 수 없었다. 본 연구에서는 비용이 적게 들고 사용자기 쉽게 사용할 수 있는 딥러닝 도구에 대한 정량적 비교 실험을 진행 하였다.

Ⅱ. 이론고찰

최근 무료로 이용 가능한 오픈소스 기반의 딥러닝 도구들이 많이 개발되고 있는데, 대표적으로 R와 텐서플로우를 들 수 있다[3]. R은 통계 처리를 목적으로 개발되었으며, 개발자들이 무료로 필요한 기능을 구현하여 패키지 형태로 제공하고 있으며, 텐서플로우는 구글에서 연구용으로 자체 개발한 도구로써, 2016년 일반인들이 사용할 수 있도록 소스코드를 개방 하였다[4]. 두 도구는 개발자들로부터 많은 각광을 받고 있으며, 사용하는 개발자와 분야도 점점 증가 추세에 있다. 기계학습을 처음 접하는 사용자들에게 딥러닝 도구 선택은 매우 어려운 문제이다[5]. 가장 중요한 이유는 직접적으로 사용해 보기 전까지는 수많은 도구들의 장단점이나 성능에 대해서 파악이 어렵다는 점이다[6]. 본 연구에서는 이러한 문제점을 해결하고 개발자들에게 도구 선택시 도움을 제공하고자 R과 텐서플로우에 대한 성능 비교 실험을 진행 하였다. 딥러닝 도구에 대한 성능 비교 연구는 일부 연구자들에 의해 진행이 되었지만, R과 텐서플로우에 대한 정량적 성능 비교 연구는 아직까지 없었다[7-10].

Ⅲ. 연구 방법

R과 텐서플로우에서 학습을 수행하기 위한 과정은 약간 차이가 존재한다. 텐서플로우의 경우, 딥러닝 모형 구축, 컴파일, 학습 수행 과정을 별도의 코드로 구현해 주어야 하며, R의 경우에는 모두 하나의 과정으로 통합되어 있다. R을 활용한 딥러닝의 경우, nnet 패키지나 neuralnet 패키지 2가지의 함수를 이용할 수 있다. nnet 패키지는 초기에 개발된 딥러닝 학습 도구로써, 은닉층을 1개만 정의할 수 있다. 일반적으로 은닉층을 2개 이상 정의할 수 있어야 딥러닝 모델이라고 할 수 있는데, 1개 밖에 정의하지 못하기 때문에, 실험에서는 사용하지 않았다. 따라서, 본 연구에서는 최근에 개발된 neuralnet 패키지를 사용해서, R기반 딥러닝 실험을 수행 하였다. 두 도구의 성능을 비교하기 위한 실험 방법은 [그림 1]과 같다. [그림 1(a)]는 텐서플로우 기반 실험 절차를 나타내고 [그림 1(b)]는 R기반 실험 절차를 나타낸다.

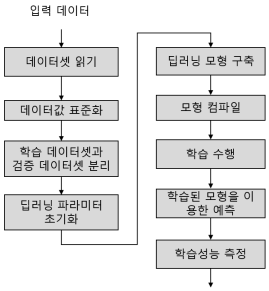

그림 1(a). 텐서플로우 성능 측정 방법
Figure 1(a). Experiment method for measuring Tensorflow performance

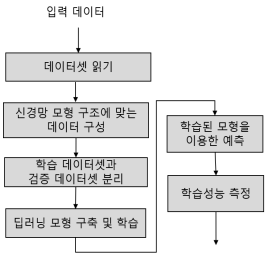

그림 1(b). R 성능 측정 방법
Figure 1(b). Experiment method for measuring R performance

텐서플로우의 경우에는 딥러닝 모형을 행렬을 이용하여 직접 구축하거나 케라스를 사용하는 방법이 있다. 첫번째 방법은 시간이 오래 걸리고 구현이 복잡하기 때문에, 본 연구에서는 케라스를 사용 하였다.

두 도구의 성능을 비교하기 위한 학습 데이터는 2013년 3월 1일부터 2023년 3월 29일까지 서울시 강남구에서 수집된 온도 데이터를 사용하였다. 수집된 데이터셋은 날짜별 온도값을 포함하고 있으며, 엑셀 파일 형태로 저장하였다. 또한, 원래의 데이터를 동일한 구조로 재구성한 후, 각 도구로 구축한 심층 신경망 모형의 학습 데이터로 사용하였다. 실험에서는 은닉층을 1개 또는 2개 가지는 6종류의 심층 신경망을 구축하였다. [표 1]은 실험에서 사용된 6종류의 심층 신경망의 내부 노드 숫자를 보여주고 있다.

표 1. 실험에 사용된 신경망 구조
Table 1. Deep neural network structures for experiment

은닉층이 1개인 경우		
입력노드 숫자	은닉층 노드숫자	출력층노드 숫자
10	5	5
10	10	5
10	15	5
은닉층이 2개인 신경망 모형		
입력노드 숫자	은닉층1	은닉층2	출력층
10	5	5	5
10	10	10	5
10	20	20	5

[그림 2]는 실험에서 사용한 심층 신경망 구조중 은닉층이 1개이고 내부 노드 숫자가 5개인 심층 신경망 구조를 보여준다.

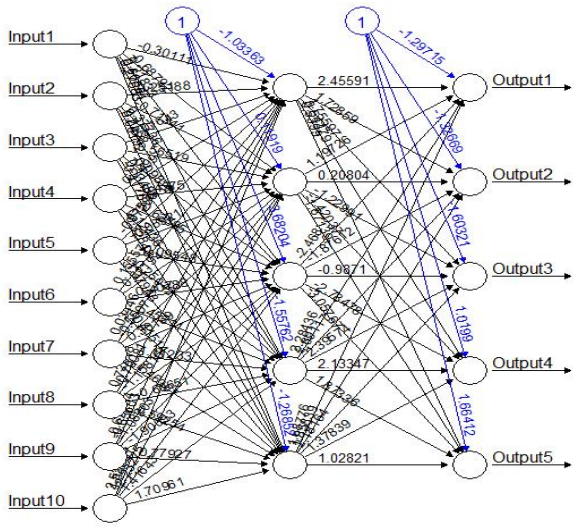

그림 2. 실험에서 사용된 심층 신경망
Figure 2. Deep neural network used in the experiment

다음절에서 각 도구를 사용한 실험 방법에 대해서 자세히 설명한다.

1. 텐서플로우 기반 딥러닝 성능 실험 방법

텐서플로우 기반 성능 실험 1단계에서는 학습 데이터셋을 구성한다. 본 연구에서는 입력 데이터셋을 csv 파일로 저장 하였으며, 온도 데이터를 메모리로 읽어온 후, 0과 1사이의 표준화된 값으로 변경하여 텐서 변수에 저장 하였다. 실험 2단계에서 0과 1사이의 값으로 데이터를 표준화 한다. 데이터 표준화 기능은 직접 작성하지 않고 텐서플로우에서 제공하는 함수를 사용하였다. 실험 3단계에서는 표준화된 데이터를 훈련 데이터와 검증 데이터로 분리 저장 한다. 이때, 원 데이터의 80%는 훈련 데이터로, 20%는 검증 데이터로 저장 하였다. 실험 4단계에서는 딥러닝을 위한 파라미터들을 초기화 한다. [표 2]는 실험에서 사용된 파라미터 이름과 파라미터 값들을 보여준다.

표 2. 심층 신경망 훈련을 위한 파라미터 값
Table 2. Parameters for Training Deep Neural Network

파라미터 이름	파라미터 값
은닉층 노드 개수	20
활성화 함수	시그모이드 함수(logistic)
반복학습수(학습주기)	30000
학습률	0.1
최적화 함수	SGD(Stochastic Gradient Descent)

[표 2]에서 서술한 파라미터 값들은 R에서 실험을 수행할 때도 동일한 값으로 설정하였다. 실험 5단계에서는 케라스 함수를 사용하여 구조의 신경망 모델을 구축 한다. 신경망 모델 구축을 위한 예제 코드는 [그림 3]과 같다.

```
k_temperature_dnn_model= Sequential()
k_temperature_dnn_model.add(Dense(20,        input_dim=10,
activation='sigmoid'))
k_temperature_dnn_model.add(Dense(20,
activation='sigmoid'))
k_temperature_dnn_model.add(Dense(5,
activation='sigmoid'))
```

그림 3. 텐서플로우를 사용한 심층 신경망 구축 코드
Figure 3. Codes for constructing deep neural network using tensorflow

위의 코드는 입력층이 10개의 노드를, 2개의 은닉층은 20개의 노드를 출력층은 5개의 노드를 가지는 딥러닝 모형을 만드는 코드이다. 만약 은닉층 노드 숫자를 5개로 변경하고 싶을 경우에는 Dense함수의 첫번째 인자값을 5로 변경하면 된다. 본 연구에서는 이 값을 바꾸어 가면서 6개의 모형을 구축하여 실험을 진행 하였다. 실험 6단계에서는 구축된 딥러닝 모델을 컴파일하고 학습을 수행 한다. 학습을 위한 학습주기는 100으로 설정하였다. 한번의 학습주기는 전체 데이터셋을 모두 사용하여 학습을 마친 시점을 의미한다. 본 연구에서는 전체 데이터셋이 약 3600건을 포함하고 있으므로, 한번의 학습 주기가 끝나면, (3000*10)번의 학습을 수행한 것이 된다. 실험 6단계에서 구축된 신경망을 학습 시킨다. 텐서플로우에서 제공하는 딥러닝 학습 함수는 fit함

수이다. 실험 7단계에서는 훈련된 신경망 모형을 이용하여 앞으로 5일간의 습도값을 예측하고 실제값과 비교하여 오차값을 계산하게 된다. 이때, 사용하는 오차 지표값으로는 평균 제곱근 오차값을 사용한다. 평균제곱근 오차값의 공식은 수식(1)과 같다.

$$\text{평균제곱근 오차(RMSE)}=\sqrt{\frac{1}{n}\sum_{j=1}^{n}(T_j-P_j)^2} \quad -(1)$$

수식 (1)에서, n은 데이터의 개수를, T_j는 j번째 데이터의 정답값을, P_j는 j번째 데이터의 예측값을 의미한다.

2. R을 사용한 성능 실험 방법

R 딥러닝 성능 실험은 총 5단계로 진행된다. 실험 1단계에서는 엑셀로 저장된 실험용 수집 데이터를 R에서 제공하는 read.xlsx2 함수를 사용하여 메모리로 읽어온 후, 데이터 프레임이라는 R의 자료구조로 저장 한다. [그림 3]은 외부에 엑셀로 저장된 온도 데이터를 엑셀로 읽어오기 위한 코드이다.

```
data <- read.xlsx2(file.choose(), 1)
```

그림 3. 훈련 데이터셋을 읽기 위한 R코드
Figure 3. R codes for reading training dataset

위 코드가 수행되고 나면, data에 온도 데이터가 저장된다. 읽어온 데이터의 내용을 확인하기 위해서는 data명령어를 입력하고 엔터키를 입력하면 된다. 변수에 저장된 데이터는 기본적으로 문자열로 저장되기 때문에, 훈련 데이터로 사용하기 위해서는 R에서 제공하는 as.numeric 함수를 이용하여 숫자로 변경하여 저장하여야 한다. 변경된 테이블의 행 이름과 열 이름은 따로 지정하지 않고 원래 파일에 저장되어 있던 열과 행 이름으로 설정하게 된다. 숫자로 변경되어 저장된 데이터는 data.frame함수를 사용하여 다시 R의 데이터 프레임 자료 구조로 변경하여 저장하다. 또한, 데이터를 0과 1사이의 값으로 표준화 한다. [그림 4]는 읽어온 데이터셋을 훈련 데이터와 검증 데이터로 분리 저장하는 코드이다.

```
train.input <- getDataSet(df$temp, 1, 3676, INPUT_NODES)
train.output <- getDataSet(df$temp, 11, 3681,OUTPUT_NODES)
data <- cbind(train.input, train.output)
colnames(data) <-c("Input1","Input2","Input3“,"Input4”
,"Input5",Input6","Input7","Input8","Input9","Input10“
,”Output1","Output2","Output3","Output4","Output5")
set.seed(1234)
n = nrow(data)
train <- sample(1:n, n*0.8,FALSE)
test <- data[-train,]
train <- data[train,]
```

그림 4. 훈련 데이터와 검증 데이터 구성을 위한 R코드
Figure 4. R codes for constructing training data and validation data

실험 2단계에서는 실험에 사용되는 신경망 구조에 맞도록 실험 데이터를 재구축 한다. R에서 제공하는 딥러닝 함수를 사용하기 위해서는 구축된 심층 신경망의 입력 노드 숫자와 동일한 숫자의 데이터를 훈련 데이터로 제공해주어야 하고, 출력 노드 숫자와 동일한 숫자의 정답 데이터를 제공해 주어야 한다. 본 연구에서는 입력 노드 숫자는 10이고 출력 노드는 5인 심층 신경망을 구축하여 사용 하였기 때문에, 10개의 훈련 데이터와 5개의 정답 데이터가 한줄의 훈련 데이터로 구성되는 3681행의 학습 데이터를 구축하였다.

실험 3단계에서는 구축된 데이터를 훈련 데이터와 검증데이터로 분리하여 저장 한다. 실험 4단계에서는 R에선 제공하는 neuralnet 함수를 사용하여 심층 신경망 모델을 구축하고 학습을 진행 하였다. 학습 함수에는 은닉층 개수, 학습률, 학습 최대횟수, 활성화 함수 등의 파라미터 값들을 설정해 주어야 한다. 여기에서 설정된 파라미터 값들은 텐서플로우와 동일한 값으로 설정하였다. [그림 5]는 구축된 신경망을 학습하는 R코드이다.

```
model <- neuralnet(Output1+Output2+Output3+
 Output4+Output5~.,
  train, hidden = 20, threshold=0.03, learningrate=0.1,
stepmax=30000, linear.output = FALSE,
  act.fct = function(x) 1/(1 + exp(-x)), # sigmoid
  err.fct = "sse" )
```

그림 5. 심층 신경망을 학습 시키는 R코드
Figure 5. R codes for training deep neural network

실험 5단계에서 예측 입력 데이터와 훈련된 신경망을 사용하여 예측을 수행한다. 예측을 수행하는데 사용하는 함수는 compute 함수이다. compute 함수의 매개변수로는 model, data 등이 있다. model에는 neuralnet 함수를 통해 학습된 신경망 모형을 넣어주어야 하며, data에는 미래값 예측을 위한 입력 데이터를 넣어 주면 된다. [그림 6]은 데이터 예측을 위한 R 코드 이다.

```
pred<-compute(model,test[,c("Input1","Input2","Input3","Input4","Input5","Input6“,"Input7","Input8","Input9","Input10")])
real<-test[,c("Output1","Output2","Output3","Output4","Output5")]
```

그림 6. 학습된 신경망을 사용한 데이터 예측 R코드
Figure 6. Prediction using a trained neural network model

실험 6단계에서 실제값과 비교하여 오차값(RMSE) 값을 구한다. 이때, 텐서플로우 실험에서 사용했던 동일한 수식(1)을 사용한다.

Ⅳ. 연구 결과

[그림 7]은 R과 텐서플로우에서 딥러닝을 수행하는 화면을 보여주고 있다. R에서는 코드작성 윈도우에 딥러닝 코드를 입력한 후, 실행키(Ctrl+R)키를 눌러서, 10번 실험을 진행 하도록 하였다. 텐서플로우의 경우, 딥러닝 과정이 화면에 표시되지만, R에서는 딥러닝 학습 과정이 화면에 표시되지는 않는다. [그림 7(a)]에서 왼쪽

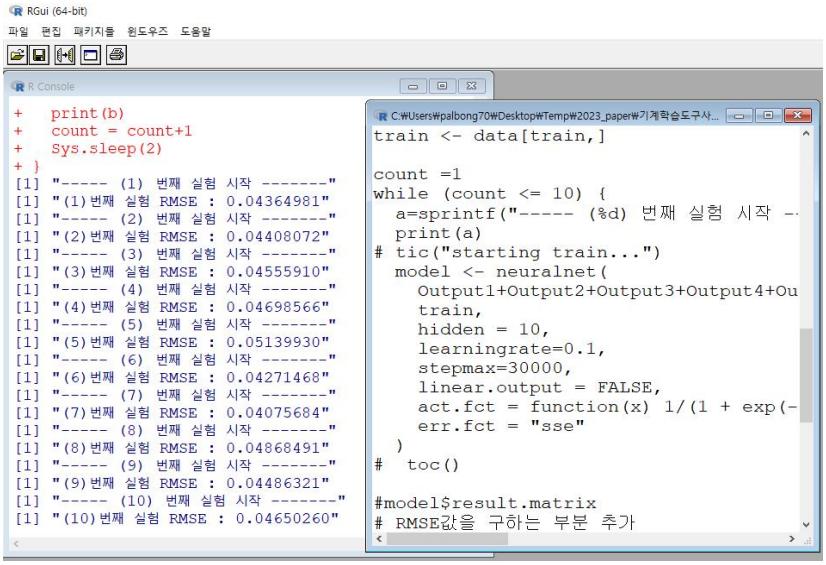

그림 7(a). R을 사용한 온도예측 딥러닝 실험 화면
Figure 7(a). Screen of deep learning experiment using R

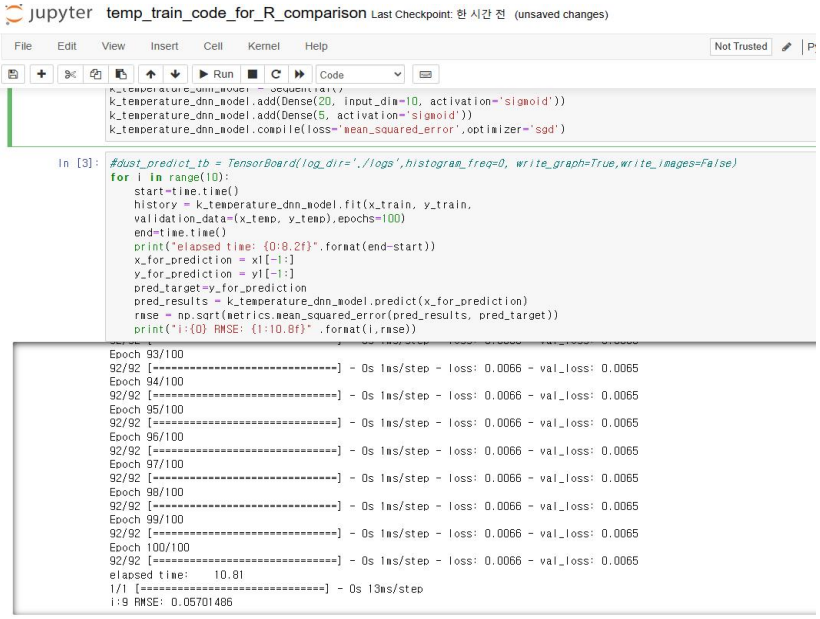

그림 7(b). 텐서플로우를 사용한 온도예측 딥러닝 실험 화면
Figure 7(b). Screen of deep learning experiment using Tensorflow

화면이 10번 반복 실험하여 RMSE를 계산한 결과값이고 오른쪽 화면이 학습을 수행하는 R코드가 포함된 스크립트 화면이다. [그림 7(b)]는 텐서플로우를 사용한 딥러닝 성능 실험 화면이다.

성능 비교를 위해, 딥러닝 수행후, 각각에 대한 RMSE값을 측정하였다. 실험 결과, R의 경우에는 은닉층이 1개인 경우, 은닉층 노드 개수를 20개로 설정한 경우, 가장 좋은 성능을 보였음을 알 수 있다. 하지만, 은닉층 노드 숫자 증가에 따른 각각의 측정값이 커다란 차이를 보이지 않기 때문에, 노드 숫자 증가에 따른 성능 향상 효과는 그다지 크지 않음을 알 수 있다. 은닉층을 2개로 늘린 경우에는 오히려 은닉층 노드 숫자를 가장 적게 설정한 경우, 오차값이 가장 적게 (값:0.04640647) 나온 것을 알 수 있으며, 노드 숫자 증가에 따른 오차값도 유의미한 차이를 보인다고 할 수 없을 정도로 작기 때문에, 은닉층이 2개인 경우, 노드 숫자에 따른 성능 차이는 크지 않다고 볼 수 있다. 또한, 전체적으로 은닉층 숫자가 1개인 경우와 2개인 경우, 오차값이 커다란 차이를 보이지 않기 때문에, 학습 시간을 고려했을 때, 은닉층 개수를 1개로 설정하여 사용하는 것이 더 좋다고 할 수 있다.

[표 3]은 실험에서 사용한 6종류의 신경망에 대한 R과 텐서플로우의 성능 측정값이다.

다음으로, 텐서플로우 결과를 살펴보자. 은닉층이 1개인

표 3. 실험 결과
Table 3. Experiment results

은닉층이 1개인 경우 오차값 측정결과(RMSE)				은닉층이 2개인 경우 오차값 측정결과(RMSE)			
은닉층 노드개수	실험 횟수	R	텐서플로우	은닉층 노드개수	실험 횟수	R	텐서플로우
5	1	0.04461693	0.06944955	(5,5)	1	0.04733705	0.05614309
	2	0.04985710	0.06695068		2	0.04973892	0.05031961
	3	0.04696456	0.06814931		3	0.04369744	0.05157458
	4	0.04629501	0.06895290		4	0.04701755	0.05436278
	5	0.04741034	0.06904894		5	0.04306604	0.05686847
	6	0.04880025	0.06837442		6	0.04862758	0.05852328
	7	0.04907759	0.06726451		7	0.04348593	0.05938368
	8	0.04387159	0.06588237		8	0.04482738	0.05943268
	9	0.04809780	0.06428031		9	0.04447521	0.05893099
	10	0.04710488	0.06271589		10	0.05179164	0.05749712
	평균	0.04720960	0.06710688		평균	0.04640647	0.05630363
10	1	0.04364981	0.07222732	(10,10)	1	0.04149925	0.06278403
	2	0.04408072	0.06758551		2	0.04553730	0.04856270
	3	0.04555910	0.06816695		3	0.04428381	0.05033989
	4	0.04698566	0.06851420		4	0.05321199	0.05323940
	5	0.05139930	0.06833293		5	0.04717330	0.05581556
	6	0.04271468	0.06762163		6	0.04340924	0.05752947
	7	0.04075684	0.06656535		7	0.05290767	0.05824149
	8	0.04868491	0.06523748		8	0.04958732	0.05825154
	9	0.04486321	0.06375411		9	0.04853829	0.05761044
	10	0.04650260	0.06229206		10	0.04695675	0.05683160
	평균	0.04832815	0.06702975		평균	0.04731049	0.05592061
20	1	0.04947739	0.05653082	(20,20)	1	0.04884166	0.06690853
	2	0.04169516	0.05879918		2	0.05114779	0.05995546
	3	0.04642122	0.06255523		3	0.05282058	0.06067481
	4	0.04445220	0.06591790		4	0.0484861	0.06183699
	5	0.04888756	0.06831440		5	0.05635266	0.06263012
	6	0.04250089	0.06978383		6	0.04882946	0.06265066
	7	0.04322166	0.07042727		7	0.04713260	0.06221826
	8	0.0491397	0.07042887		8	0.05056388	0.06132379
	9	0.04748775	0.06992780		9	0.04523569	0.06030046
	10	0.05069191	0.05701486		10	0.04806019	0.05897659
	평균	0.04639754	0.06617916		평균	0.04974706	0.06174757
전체평균(은닉층1개)		0.04731176	0.06677193	전체평균(은닉층2개)		0.04782134	0.05799060

경우 평균 오차가 약 0.06677193이고 은닉층이 2개인 경우 0.05799060로써, 은닉층이 2개인 경우가 10% 이상의 성능향상을 보이고 있음을 알 수있다. 흥미로운 점은 은닉층이 1개인 경우에는 은닉층 내부 노드 숫자를 2배씩 늘려도 오차값이 별로 차이가 없게 측정되어, 성능향상에는 별로 도움이 되지 않았던 반면, 은닉층을 2개로 늘린 경우에는 은닉층 내부 노드 숫자를 20개로 가장 크게 설정했을 때, 오차값이 가장 크게 측정되어, 가장 좋지 않은 성능을 보였다. 보통, 노드 숫자를 늘리면, 시간은 많이 걸리고 성능은 나빠지지 않는다고 알고 있었으나, 이 경우는 의외의 결과를 보이고 있음을 알 수 있으며, 향후, 원인 분석이 필요할 것으로 보인다.

전체적으로 비교해 보았을 때, 은닉층이 1개인 경우, R에서의 학습 오차 평균은 0.04731176, 텐서플로우는 0.06677193으로 측정되었으며, 은닉층이 2개인 경우에는 R이 0.04782134, 텐서플로우는 0.05799060로 측정되었다. 전체적으로 R이 성능이 더 우수한 보이고 있음을 알 수 있다. [그림 8]은 은닉층 노드 개수에 따른 오차를 비교한 것이다.

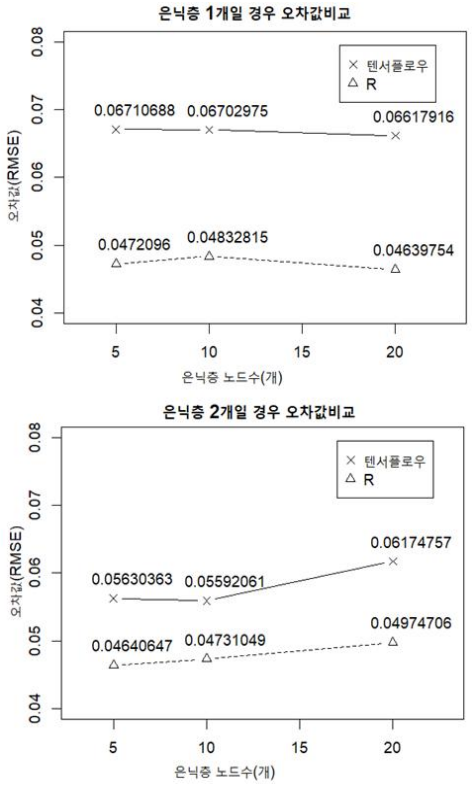

그림 8. 은닉층 개수에 따른 R과 텐서플로우 오차값
Figure 8. Error comparison according to the number of hidden nodes

V. 결 론

본 연구에서는 R와 텐서플로우에 대한 정량적 성능 평가를 진행하였다. 평가를 위해, 6종류의 심층 신경망을 구축하고 국내에서 수집된 3681건의 온도 데이터셋을 사용하여 구축된 신경망을 학습 시켰다. 학습된 신경망을 사용하고 5일간의 온도를 예측하고 오차값을 측정하였다. 측정 결과, R이 텐서플로우에 비해 성능이 우수한 것으로 측정되었다. 본 연구에서는 R이 더 우수한 성능을 보였지만, 학습 데이터셋이 달라지거나 최적화 알고리즘을 다르게 설정할 경우, 결론이 달라질 수도 있을 것으로 생각된다. 이 부분은 추후에 좀더 연구가 되어야 할 것으로 생각된다.

References

[1] R.V.D. Schoot, J.D. Bruin, R. Schram, P. Zahedi, J.D. Boer, F. Weijdema, B. Kramer, M. Huijts, M. Hoogerwerf, G. Ferdinands, A. Harkema, J. Willemsen, Y. Ma, Q. Fang, S. Hindriks, L. Tummers, D. L. Oberski, “An open source machine learning framework for efficient and transparent systematic reviews,” *Nature Machine Intelligence*, Vol. 3, pp. 125 - 133, 2021(https://doi.org/10.1038/s42256-020-00287-7)

[2] Y.G. Yoon and T.G Oh, “A Study on the Improvement of Construction Site Worker Detection Performance Using YOLOv5 and Open Pose,” *The Journal of the Convergence on Culture Technology (JCCT)*, Vol. 8, No. 5, pp.735-740, 2022(DOI: http://dx.doi.org/10.17703/JCCT.2022.8.5.735)

[3] P. Sherkhane and D. Vora, “Survey of deep learning software tools,” *2017 International Conference on Data Management, Analytics and Innovation (ICDMAI),* pp. 24-26, 2017 February, Pune, India.

[4] Z. Pala, “Examining EMF Time Series Using Prediction Algorithms With R,” *IEEE Canadian Journal of Electrical and Computer Engineering*, Vol.44, No.2, 2021, pp. 223-227 (https://10.1109/ICJECE.2020.3037805)

[5] Y.-S. Lee and P.J. Moon, “A Comparison and Analysis of Deep Learning Framework,” *The Journal of the Korea Institute of Electronic Communication Sciences*, Vol.12, No.1, pp. 115-122, 2017((https://doi.org/10.13067/JKIECS.2017.12.

1.115.)
[6] J.H. Jang , J.H. Park, H.J. Kim, S. R. Yoon, “A Comparative Performance Analysis of Spark-Based Distributed Deep-Learning Frameworks,” *KIISE transactions on computing practices*, Vol.23 No.5, pp. 299-303, 2017(DOI:https://doi.org/10.5626/KTCP.2017.23.5.299)
[7] L. Ferreira, A. Pilastri, C. M. Martins, P. M. Pires, P. Cortez, “A Comparison of AutoML Tools for Machine Learning, Deep Learning and XGBoost,” *2021 International Joint Conference on Neural Networks (IJCNN)*, July 2021(https://10.1109/IJCNN52387.2021.9534091)
[8] Y.-S. Lee, “Comparison of Machine Learning Tools for Mobile Application,” *International Journal of Advanced Culture Technology(IJACT)*, Vol. 10 No.3, pp.360-370, 2022(https://doi.org/10.17703/IJACT.2022.10.3.360)
[9] S.A. Ali, “LOsMonitor: A Machine Learning Tool for Analyzing and Monitoring Cognitive Levels of Assessment Questions,” *IEEE Transactions on Learning Technologies,* Vol. 14, No.5, pp.640-652, 2021(https://doi.org/10.1109/TLT.2021.3116952)
[10]C. Francisco, “A Comprehensive and Didactic Review on Multilabel Learning Software Tools,” *IEEE Access*, Vol. 8, pp. 50330-50354, 2020(DOI:https://doi.org/10.1109/ACCESS.2020.2979787)

※ 본 연구는 금오공과대학교 교수연구년제에 의하여 연구된 실적물임.