Journal of Digital Convergence
Vol. 18. No. 11, pp. 259-266, 2020

ISSN 2713-6434 / eISSN 2713-6442
https://doi.org/10.14400/JDC.2020.18.11.259

LSTM을 이용한 주가예측 모델의 학습방법에 따른 성능분석

정종진[1], 김지연[2*]
[1]대진대학교 휴먼IT융합학부 교수, [2]대진대학교 창의미래인재대학 교수

A Performance Analysis by Adjusting Learning Methods in Stock Price Prediction Model Using LSTM

Jongjin Jung[1], Jiyeon Kim[2*]
[1]Professor, Division of Human IT Convergence, Daejin University
[2]Professor, College of Humanities and Arts, Daejin University

요 약 과거 인공지능 분야에서는 지식 기반의 전문가 시스템 및 머신러닝 알고리즘들을 금융 분야에 적용하는 연구가 꾸준하게 수행되어 왔다. 특히 주식에 대한 지식 기반의 시스템 트레이딩은 이제 보편화되었고, 최근에는 대용량 데이터에 기반한 딥러닝 기술을 주가 예측에 적용하기 시작했다. 이중 LSTM은 시계열 데이터에 대한 검증된 모델로서 주가 예측에도 적용되고 있다. 본 논문에서는 주가 예측 모델로서 LSTM을 적용할 때 성능향상을 위해 고려해야 할 복잡한 매개변수 설정과 적용 함수들에 대해 적합한 조합 방법을 제안하도록 한다. 크게 가중치와 바이어스에 대한 초기화 대상과 설정 방법, 과적합을 피하기 위한 정규화 적용 대상과 설정 방법, 활성화 함수 적용 방법, 최적화 알고리즘 선택 등을 제시한다. 이 때 나스닥 상장사들에 대한 대용량 데이터를 바탕으로 각각의 방법들을 적용하여 정확도를 비교하면서 평가한다. 이를 통해 주가 예측을 위한 LSTM 적용 시 최적의 모델링 방법을 실증적인 형태로 제안하여 현실적인 시사점을 갖도록 한다. 향후에는 입력 데이터의 포맷과 길이, 하이퍼파라미터들에 대한 성능평가를 추가 수행하여 주요 설정 항목들의 조합에 대한 일반화 연구를 수행하고자 한다.

주제어 : LSTM, 딥러닝, 주가, 하이퍼파라미터, 예측 모델

Abstract Many developments have been steadily carried out by researchers with applying knowledge-based expert system or machine learning algorithms to the financial field. In particular, it is now common to perform knowledge based system trading in using stock prices. Recently, deep learning technologies have been applied to real fields of stock trading marketplace as GPU performance and large scaled data have been supported enough. Especially, LSTM has been tried to apply to stock price prediction because of its compatibility for time series data. In this paper, we implement stock price prediction using LSTM. In modeling of LSTM, we propose a fitness combination of model parameters and activation functions for best performance. Specifically, we propose suitable selection methods of initializers of weights and bias, regularizers to avoid over-fitting, activation functions and optimization methods. We also compare model performances according to the different selections of the above important modeling considering factors on the real-world stock price data of global major companies. Finally, our experimental work brings a fitness method of applying LSTM model to stock price prediction.

Key Words : LSTM, Stock Price, Deep Learning, Hyper-parameter, Prediction Model

*This research was results of a study on the "HPC Support" Project, supported by the 'Ministry of Science and ICT' and NIPA.
*Corresponding Author : Jiyeon Kim(jykim629@daejin.ac.kr)
Received August 25, 2020 Revised September 26, 2020
Accepted November 20, 2020 Published November 28, 2020

1. 서론

금융 분야에서의 주가 예측은 인공지능의 도전 문제 중 하나이다. 과거에는 주로 지식 기반의 전문가 시스템 접근방법이나 전문가 시스템에 주가 예측의 모호성에 대한 퍼지니스(Fuzziness)를 결합하는 방법, 또는 전통적인 통계 기법에 기초한 방법 등을 적용하였다 [1-3]. 뿐만 아니라 주식 분야의 데이터에 기반으로 하는 머신러닝 기법들도 적용되어 왔다. 머신러닝 기법으로는 주로 SVM(Support Vector Machine)이나 MLP(Multi-Layered Perceptron) 등이 적용되어 왔다[4-6]. 예를 들어, [5]에서는 회사의 재무정보를 SVM에 학습하여 주가를 예측하는 연구를 수행하였고, [6]에서는 kNN과 SVM을 사용하여 주가 예측하는 시스템을 개발하였다. 이 연구에서는 프렉탈, 모멘텀, 변동성의 특징을 추출하고, 이 후 kNN과 SVM을 활용한 모델을 만들어 다음 달의 지수를 예측하는 방법을 구현하였다. 이 때 알고리즘의 파라미터 최적화를 위해 kNN에 대해서는 k값과 SVM에 대해서는 RBF 커널의 정규화 관련 파라미터인 C값에 따른 정확도 변화를 측정하였다. [7]은 감성 분석을 통해 주가의 움직임을 예측하려 했는데 1000만 여개의 트위터 감성을 Calm, Alert, Sure, Vital, Kind, Happy 등의 6가지 감성으로 분류하고, 이를 사용하여 미국 다우존스의 등락을 예측하였다. 최근에는 머신러닝의 심화학습 모델들을 다루는 딥러닝을 이용한 연구들도 활성화되고 있다. 기존에는 대용량 데이터의 학습에 필요한 하드웨어 환경의 진입장벽으로 인해 연구가 어려웠지만 최근에는 이 문제가 제약이 되지 않고 기존 모델들의 개선 기법들이 등장하게 됨에 따라 딥러닝을 적용하여 예측 정확도가 크게 향상되고 있는 것이다[8-13]. [10]에서는 날짜, 종가, 상한가, 하한가, 시작가, 거래량과 같은 1차원 시계열 데이터를 2차원 그래프로 변환하여 CNN 모델의 입력값으로 사용하여 주가를 예측한다. [11]에서는 MLP, CNN, RNN의 세 가지 딥러닝 모델이 예측한 결과를 결합하고 MLP를 사용하여 다시 학습하는 스태킹(Stacking) 기반의 앙상블 모델을 사용하여 주가를 예측한다.

본 연구에서는 딥러닝 모델인 LSTM(Long Short Term Memory)을 이용한 주가 예측 모델을 구현하고 모델 성능에 영향을 미치는 중요한 학습 방법들에 대해 실험하였다. LSTM은 기존의 RNN(Recurrent Neural Network) 모델이 학습이 길어지면 초기에 학습한 결과를 잊어버리는 기울기 소실(Vanishing Gradient) 문제를 극복하기 위해 입력 게이트, 출력 게이트, 망각 게이트로 구성된 셀(Cell)을 추가해서 개선한 모델로서, 과거 학습정보를 기억하고 새로운 학습결과에 반영이 가능해서 시계열 문제 및 예측 문제에 성능을 발휘하는 학습 모델이다. 따라서 본 논문에서는 전형적인 시계열 데이터인 주가 데이터에 대해 최근에 활용성이 더욱 높아지고 있는 LSTM 모델을 적용하는 방법론을 제시한다는 측면에서 의의를 두고 연구를 수행하였다.

2. LSTM 구조와 동작

LSTM은 Fig. 1과 같이 RNN(Recurrent Neural Network)의 은닉층(Hidden Layer)에 셀 상태(Cell State)를 추가해서 개선한 모델로서, 과거 학습 정보를 기억하고 새로운 학습 결과에 반영할 수 있게 된다. 따라서 LSTM은 시계열 문제 및 예측 문제에 뛰어난 성능을 발휘하는 학습 모델이라고 할 수 있다.

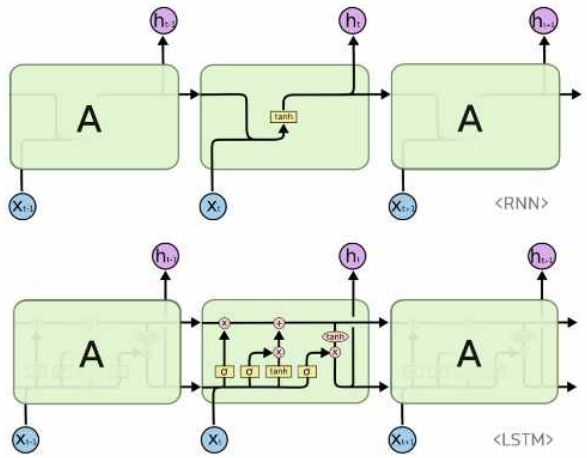

Fig. 1. Architecture of LSTM

셀의 수식은 다음과 같다.

$$ft = \sigma(Wxh\_fxt + Whh\_fht-1 + bh\_f)$$
$$it = \sigma(Wxh\_ixt + Whh\_iht-1 + bh\_i)$$
$$ot = \sigma(Wxh\_oxt + Whh\_oht-1 + bh\_o)$$
$$gt = \tanh(Wxh\_gxt + Whh\_ght-1 + bh\_g)$$
$$ct = ft \odot ct-1 + it \odot gt)$$
$$ht = ot \odot \tanh(ct)$$

Fig. 2는 셀 상태를 포함하는 LSTM 블록을 확대한 구조의 모습이다 [14].

Fig. 2에서 보이는 것처럼, LSTM 블록은 입력 게이트(Input Gate), 망각 게이트(Forget Gate), 출력 게이트(Output Gate)라는 3가지 게이트가 붙어 있는 셀 상태

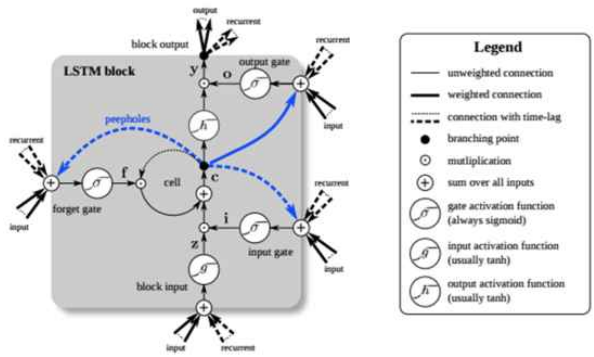

Fig. 2. Internal block architecture of LSTM

로 구성되어 있다. 입력 게이트(it ⊙ gt)는 현재의 정보를 기억하기 위한 게이트로서, 입력(input) xt와 순환(recurrent) ht−1을 받아서 시그모이드(σ)를 취하고 같은 입력 xt에 대해 tanh를 취한 다음 hadamard product 연산해서 내보낸다. 망각 게이트(ft)는 과거 정보를 잊기 위한 게이트로서, 입력 xt와 순환 ht−1을 받아서 시그모이드를 취해서 내보낸다. 이 때 시그모이드 함수의 출력 범위가 0 ~ 1 사이이므로 그 값이 0이면 이전 상태의 값은 잊고 1이면 이전 상태의 값을 기억하게 된다. 각각의 게이트들은 0 ~ 1 사이의 값을 가지게 되고 셀은 연결되어 있는 게이트들의 값을 체크하면서 데이터를 불러올지, 유지할지, 내보낼지를 결정하게 된다. 또한 게이트에 포함되는 가중치(W)와 바이어스(b)는 학습해야 할 대상이다. 본 연구에서는 딥러닝 모델인 LSTM을 이용한 주가 예측 모델을 Keras[15]를 이용하여 구현하고 Keras 기반 LSTM 모델 적용 시에 모델 성능에 영향을 미치는 중요한 매개변수들에 대해 실험하였다.

3. 주가 예측 모델 구축 및 평가

3.1 데이터 수집 및 전처리

본 연구에서는 미국의 대표적인 글로벌 회사들에 대한 주가 데이터를 바탕으로 LSTM 기반의 예측 모델을 수립하였다. 이 때 학습 및 예측하는데 있어서 필요한 주가 히스토리 데이터가 준비되어야 하는데, 이는 Yahoo API[16]를 통해 수집하였다. 원시 데이터는 날짜, 시작가, 최고가, 최저가, 종가, 수정종가, 거래량으로 구성되어 있다. 사용한 데이터는 2018년 11월 1일 기준 나스닥의 상위 종목 20개 중 훈련(Training)에 10종목, 예측(Prediction)에 5종목을 사용했으며 데이터는 2000년 1월 1일부터 2018년 11월 중순까지의 데이터를 사용했다. 이에 따라 훈련에 사용한 종목은 애플, 어도비, 아마존, 컴캐스트, 코스트코, CISCO, 인텔, 마이크로소프트, 엔비디아, 펩시 등이며, 예측에 사용한 종목은 구글, 페이스북, 페이팔, 브로드컴, 퀄컴 등이다. 이들을 선택한 이유는 동일한 주식시장의 상장종목이며 가격 단위가 미국 달러로 일정하기 때문이다. 데이터에 대한 전처리 작업으로서 스케일링(Scaling)을 해야 하는데 그 이유는 이 회사들의 주가 범위가 크게 다르므로 데이터의 스케일을 맞춰 가중치의 스케일도 일관성 있게 맞춰지는 효과를 위해서이다. 사용한 방식은 데이터의 최대값을 1, 최소값을 0으로 두는 최소최대(MinMax) 스케일링 방법이다.

학습을 위한 데이터의 특징(Feature)을 선택할 때, 본 모델은 다음날의 수정종가를 예상하기 위한 모델이기 때문에 시작가, 최고가, 최저가, 종가를 사용하지 않고 수정종가를 선택하였고, 거래량은 해당 종목의 예측에 주로 쓰이는 데이터 중 하나이기 때문에 특징값으로 사용하였다. 또한 값이 변화하는 경향을 파악하기 위해 전날 데이터와 비교한 수정종가 변화량과 거래량 변화량을 특징값으로 사용하였다. 그런데 거래량 특징값에 대해서는 거래량 수치와 경향(거래량의 변화량)을 같이 학습시켜야 하는지, 아니면 경향만 학습시키면 되는 것인지가 검토 과정에서 문제가 되었다. Fig. 3에서 보이는 바와 같이 실제 학습을 수행해서 모델의 성능을 평균제곱오차(RMSE) 방법으로 비교한 결과, 경향만 있는 경우가 더 성능이 좋아서 모델에서 거래량 수치를 제외시켰다.

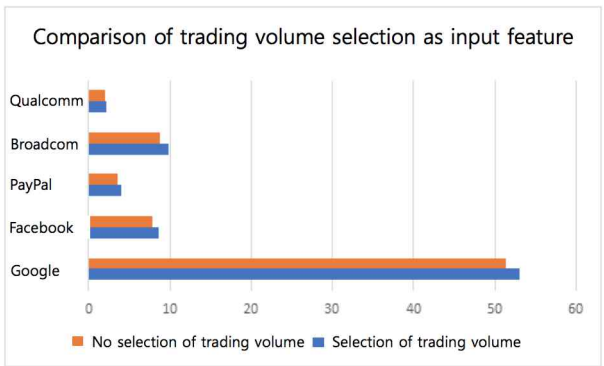

Fig. 3. Comparison of input feature selections

구성한 학습 데이터 셋은 학습 그리고 자체적인 평가를 위해서 훈련(Train), 검증(Validation), 추론(Test) 데이터 셋으로 나누게 되는데, 훈련 데이터 셋은 학습을 위한 것이고, 검증 데이터 셋은 중간 학습 평가 및 학습률(Learning Rate) 등의 파라미터 값을 수정하기 위한 것, 추론 데이터 셋은 학습에 대한 최종 정확도를 측정하기

위한 것이다. 데이터 셋을 이러한 3가지 서브 셋으로 나누게 되는데 그 비율은 모델마다 다르고, 데이터 셋의 유형에 따라 다르게 세팅된다. 주가 데이터는 시계열적 특성을 가지기 때문에 훈련 데이터 셋과 추론 데이터 셋을 나눌 때 주의해야 한다. 예를 들어, 다음의 Fig. 4처럼 아마존의 주가 흐름을 보면 2000년부터 2015년까지는 주가 변동 폭이 완만하게 증가하지만, 2016년부터는 변동 폭이 급격하게 증가하는 것을 볼 수 있다. 따라서 시간 순으로 나열되어 있는 전체 데이터 셋을 단순히 7:3의 비율로 학습 데이터 셋과 추론용 데이터 셋으로 나누게 된다면 두 데이터 셋의 경향이 다르기 때문에 올바른 학습이 이루어질 수 없게 되어 예측 정확도가 낮아지게 될 것이다[11].

Fig. 4. Amazon stock price flow by year

이를 해결하기 위해서는 Fig. 5와 같이 전체 데이터 셋을 시간 순으로 30%, 30%, 40%의 세 셋으로 먼저 분할한다. 그런 후에 각 셋을 다시 학습용 70%, 검증용 10%, 추론용 20%로 사용한다. 최종 성능은 세 셋의 평균값으로 한다.

Fig. 5. Split a time series data set

3.2 학습 모델 구축 및 평가

본 연구에서는 주가 예측 모델로 LSTM을 적용하는데 있어서 Fig. 6에 표시된 것처럼 모델의 성능에 중요한 영향을 미치는 주요 학습 방법들을 어떤 형태로 정해야 최적의 성능을 발휘할 수 있는지를 실험적으로 알아보았다. 이를 위해 주가 예측 모델을 Keras 환경에서 구현하는데 있어서 정해야 할 주요 학습방법들에 대해서 각각 해당하는 매개변수 항목들을 설정하고 이들을 복합적으로 조정하면서 최적의 조건을 찾는 실험을 수행하였다.

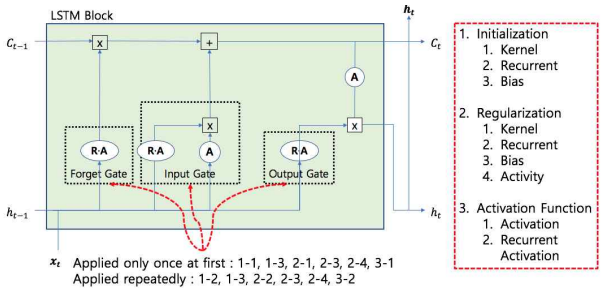

Fig. 6. LSTM structure and parameter adjustment items

본 모델의 성능향상을 위한 매개변수 조정 항목들은 크게 초기화(Initialization) 방법, 정규화(Regularization) 요소, 활성화 함수(Activation Function)로 구분되고 각각의 항목들은 다시 세분화되어 설정함에 따라 예측 정확도에 영향을 미치게 된다. 아울러, 세부 평가항목 별로 학습 시 첫 1회만 적용하는 항목들과 이후에 적용하여 학습되는 항목들로 구성된다. 또한 매개변수 조정에 따른 모델의 성능평가 방법은 앞에서 언급한 것처럼 평균제곱근오차(RMSE: Root Mean Square Error)를 사용하였다. 이 방법은 모델의 예측값과 실제 환경에서 관찰되는 값의 차이를 다룰 때 흔히 사용하는 평가방법이다. 본 논문에서 사용하는 데이터 셋은 글로벌 회사들의 실제 발생한 주가를 대상으로 하고 있으므로 RMSE를 통해 쉽고 정확하게 성능평가를 수행할 수 있으므로 선택했다. 다음 절부터는 각각의 평가항목들의 의미와 이를 적용한 모델 수행결과들에 대해 분석한다.

3.3 초기화(Initialization) 방법

딥러닝 모델에서 가중치의 초기화 방법이 모델 성능에 중요한 영향을 미친다는 것은 이미 알려진 사실로서 여러 가지 초기화 방법들이 연구되어 적용되고 있다. LSTM의 초기화에는 kernel matrix의 가중치를 초기화하는 kernel initializer, recurrent matrix의 가중치를 초기화하는 recurrent initializer, 바이어스 벡터(Bias Vector)를 초기화하는 bias initializer가 있다. 본 연구의 구현 및 실험 환경인 Keras에서 각각의 초기화 방법

에는 텐서(Tensor)를 0으로 초기화하는 zeros, 텐서를 1로 초기화하는 ones, 텐서를 일정한 값으로 초기화하는 constant, 텐서를 정규분포로 생성하는 random normal, 텐서를 균일분포로 생성하는 random uniform, 텐서를 잘린 정규분포로 생성하는 truncated normal 등이 있다. 또한 스케일링을 가중치의 형태(Shape)에 맞게 조절할 수 있는 variance scaling, 렌덤 직교행렬을 생성하는 orthogonal, 단위행렬을 생성하는 identity, LeCun이 제안한 LeCun uniform initializer, LeCun normal initializer, Xavier가 제안한 Xavier normal initializer(=glorot normal), Xavier uniform initializer(=glorot uniform), He가 제안한 He normal initializer, He uniform initializer를 사용할 수 있다. 이 때 normal과 uniform은 각각 정규분포와 균일분포의 형태를 의미한다. 본 논문에서는 이러한 초기화 방법들을 각각 적용하여 성능평가를 수행하였다. 이 때 앞 절에서 언급한대로 훈련에 사용한 종목은 애플을 비롯한 10개 종목에 대해 훈련 및 추론 과정을 거친 후, 실제 예측은 구글, 페이스북, 페이팔, 브로드컴, 퀄컴 등의 5개 종목에 대해 수행하였다. 그 결과 먼저 kernel initializer에 대한 실험에서 RMSE가 낮은 방법들은 random normal, random uniform, xavier normal, xavier uniform, lecun normal이었다. recurrent initializer에 대한 실험에서 RMSE가 낮은 방법들은 constant, ones, random normal이었다. bias initializer에서 RMSE가 낮은 방법들은 he uniform, random uniform, truncated normal이었다. 그러나 실제 LSTM 모델을 사용할 때에는 성능을 높이기 위해서 위의 세 가지 초기화 방법들을 모두 적용해야 하므로 이들의 조합 형태로 다시 실험하여 의미 있는 결과를 도출하였다.

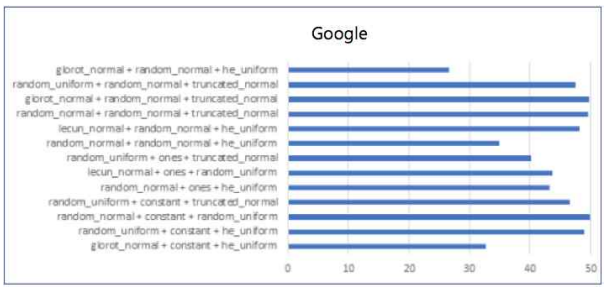

Fig. 7. The result of setting initializer

위의 Fig. 7은 구글 주가에 대해 예측을 수행하였을 때 초기화 조합을 적용한 결과를 보여주고 있다. 예측에 사용한 5개 종목 모두 가장 효과가 좋았던 것은 kernel initializer에는 Xavier normal(glorot normal), recurrent initializer에는 random normal, bias initializer에는 he uniform 방법을 적용한 조합이었다.

3.4 정규화(Regularization) 방법

LSTM을 포함한 딥러닝 모델에서 과적합(Over-fitting)을 방지하기 위한 몇 가지 방법이 있으나 가장 중요한 요소는 정규화이다. 정규화는 모델의 복잡도를 낮추기 위해 가중치의 비중을 줄여주는 개념이다. LSTM에 적용할 수 있는 정규화 대상으로는 kernel regularizer, recurrent regularizer, bias regularizer, activity regularizer가 있고, 각각에 대해서 L1 정규화 혹은 L2 정규화를 적용할 수 있다. 본 논문에서 여러 가지 실험을 수행해본 결과, kernel에 대해서는 L1을 적용할 때 유의미하게 성능이 매우 나빠졌고, L2를 적용할 때 성능이 유의미하게 좋아지는 것을 확인할 수 있었다. 또한 recurrent에 대해서는 L1, L2 모두 유의미한 변화를 보이지 않았고, bias에 대해서는 L1과 L2 모두 유의미하게 성능이 나빠졌다. 또한 activity에 L1을 적용할 때 유의미하게 성능이 매우 나빠졌고 L2를 적용할 때 유의미하게 성능이 좋아졌다.

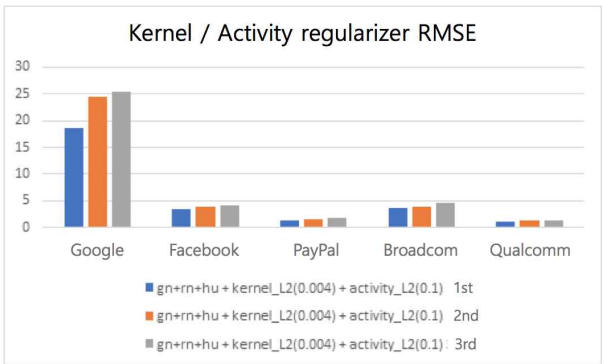

Fig. 8. The result of setting kernel/activity regularizer

Fig. 8과 같이, 정규화에 대해서도 유의미한 성능 개선을 보인 kernel regularizer와 activity regularizer를 조합해 실험하였다. 이때 activity는 L2(0.1)로 고정한 상태에서 kernel을 각각 L2(0.003)와 L2(0.004)로 설정하여 실험하였다. 결과적으로 kernel이 L2(0.003)일 때가 약간 성능이 좋아졌다.

3.5 활성화(Activation) 함수

LSTM 셀(Cell) 활성화 함수는 LSTM 층에서 선택 가능하고, 기본 활성화(Activation) 함수와 순환 활성화(Recurrent Activation) 함수 등 2가지 활성화 함수를 사용한다. 여기에는 딥러닝 분야에서 연구되어 적용되고 있는 다양한 함수들을 사용 가능하고, softmax, elu, selu, softplus, softsign, relu, tanh, hard sigmoid 등이 있다. 실험 결과, 기본 활성화 함수로는 elu 또는 softsign, tanh가 주로 좋은 성능을 보였고, recurrent 활성화 함수로는 elu, softsign, tanh, hard sigmoid가 좋은 성능을 보였다. 이에 따라 각각 좋은 성능을 보였던 활성화 함수들의 조합을 사용해서 다시 시도해 보았다.

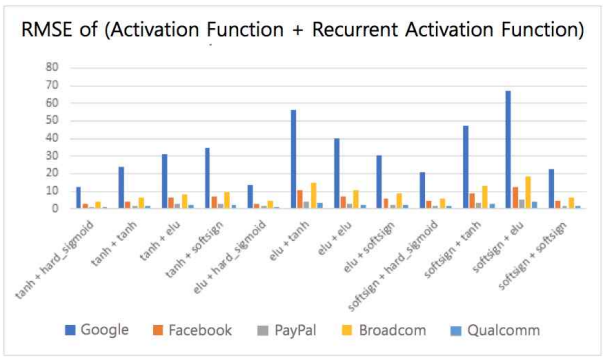

Fig. 9. The result of applying activation functions

Fig. 9에서 보이듯이, 기본 활성화 함수와 순환 활성화 함수의 조합으로 기본 설정인 (tanh, hard sigmoid)가 가장 성능이 좋았으며 아주 근소한 차이로 (elu, hard sigmoid)가 두 번째로 성능이 좋았다.

3.6 최적화(Optimization) 함수

최적화 함수는 모델의 가중치 학습 시 적용하는 알고리즘이다. 모델이 훈련 데이터를 가지고 학습을 진행하면서 데이터에 최적화된 가중치들을 탐색하는데 모델의 계산값과 결과값(레이블)의 차이를 정의하는 손실 함수(Loss Function) 값을 최소화하기 위해 기본적으로 기울기 감소(Gradient Descent) 방법을 사용한다. 그런데 기울기 감소 방법은 최소값을 찾기 위한 최적화 경로를 탐색하는 과정에서 최소값 수렴 여부와 계산속도, 국부 최적해(Local Minimum) 등 문제점이 있으므로 이를 개선한 여러 가지 최적화 함수들이 연구되어 적용되고 있다. 본 연구의 구현 및 실험 환경인 Keras에서는 대표적인 최적화 함수들인 SGD, RMSProp, Adagrad, Adadelta, Adam, Adamax, Nadam 등 7가지를 적용할 수 있다. 먼저 SGD(Stocastic Gradient Descent)는 확률적 경사하강법으로 미분을 통해 기울기를 구하여 가중치를 갱신하는 방법으로서, 훈련 데이터 셋 전체(Batch)에 대해 손실 함수를 계산하는 대신 미니 배치(Mini-batch)의 작은 크기로 계산하고 이 작업을 반복 수행함으로써 계산 속도를 개선하고 국부 최적해에 빠질 위험성을 줄이는 방법이다. 다른 방법들은 SGD 방법을 개선한 변형 방법들이다. momentum은 SGD의 최적화 경로를 탐색할 때 방향에 따라 기울기가 달라지는 비등방성 함수의 경우 탐색 경로가 지그재그로 이동하게 되는 진동이 생겨 비효율적으로 탐색하는 문제를 개선하기 위해 탐색 경로의 이동과정에 물리학적인 속도 개념을 도입하여 일종의 관성을 부여하는 방법이다. 이 방법은 현재의 기울기 감소 방향으로의 이동 벡터를 계산할 때 이전 이동 벡터를 기억하고 있다가 현재의 이동 벡터에 이전의 이동 벡터를 관성의 힘만큼 추가적으로 더하면서 이동하게 함으로써 진동부분을 완화시키는 방법이다. Nesterov momentum은 momentum의 관성 방향의 이동량을 제어함으로써 효과적으로 이동할 수 있도록 하는 방법이다. Keras에서 SGD를 선택할 경우 “keras.optimizers.SGD (lr=0.01, momentum=0.0, decay=0.0, nesterov=False)”과 같이 momentum과 Nesterov momentum에 대한 설정을 포함하여 실행할 수 있다. Adagrad는 학습을 진행하면서 점차적으로 학습률을 줄여나가는 방법으로서, 이전 기울기 값을 제곱해서 계속 더하는 방식으로 학습률을 낮춘다. RMSProp과 AdaDelta(Adaptive Delta)는 Adagrad가 기울기값의 제곱만큼 학습률을 점점 줄여나가므로 학습이 오래 진행될 경우 학습이 거의 되지 않는다는 단점을 극복하기 위하여 학습률을 낮출 때 이전 기울기 제곱의 합이 아닌 지수평균으로 학습률을 낮추는 방법이다. Adam은 학습률을 줄여나가는데 있어서 momentum의 개념처럼 속도를 계산하여 학습률을 줄이는 정도를 적응적으로 조정해 나가는 방법이다. Adamax와 Nadam은 Adam을 변형한 방법으로서, Nadam은 Adam RMSprop with Nesterov momentum이다. 본 연구에서는 활성화 함수 조합에서 성능이 잘 나온 (tanh, hard sigmoid)와 (elu, hard sigmoid)에 대해 최적화 함수를 각각 실험하였다.

실험 결과, Fig. 10에서 보이는 바와 같이, 활성화 함수가 tanh일 때는 최적화 함수가 Adam일 경우가 가장 성능이 좋았고 RMSProp, Adamax, Nadam이 유의미한 성능 개선을 보였다. 또한 Fig. 11에서 보이는 바와

같이, 활성화 함수가 elu일 때는 최적화 함수가 Adamax 일 경우가 가장 성능이 좋았으며 Adam이 비슷한 성능을 보였다.

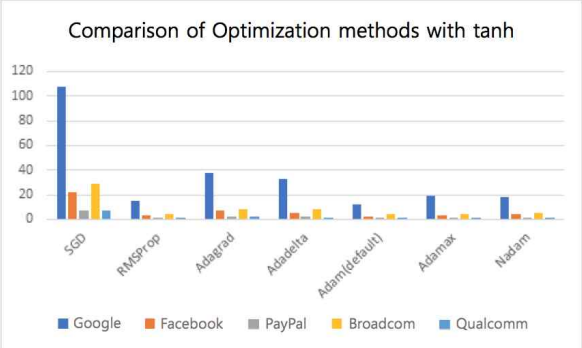

Fig. 10. The result of applying optimization function 1

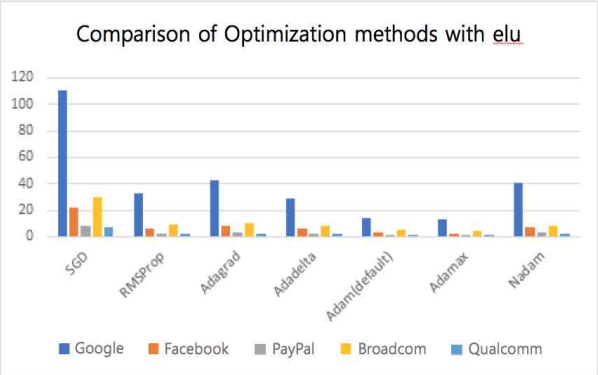

Fig. 11. The result of applying optimization function 2

4. 실험 및 평가 환경

본 연구에서 적용한 LSTM 모델을 구동한 서버의 하드웨어 환경으로는 CPU intel i7 6700, RAM 16 GIB, SSD 256GB Samsung evo pro 850, VGA GTX 1080을 탑재한 서버이고, 사용된 운영체제는 Ubuntu 16.04, 사용된 딥러닝 라이브러리 및 소프트웨어는 Tensorflow의 고수준 버전인 Keras, CUDA 8.0, Cudnn 6.0 등이 사용되었다.

학습 및 예측하는데 있어서 주가 데이터는 3.1절에서 설명한 바와 같이, 나스닥의 상위 대표 종목들 15개를 선택하고 이 중 훈련(Training)에 10종목, 예측(Prediction)에 5종목을 사용하였다. 이에 따라 훈련에 사용한 종목은 애플, 어도비, 아마존, 컴캐스트, 코스트코, CISCO, 인텔, 마이크로소프트, 엔비디아, 펩시 등이며, 예측에 사용한 종목은 구글, 페이스북, 페이팔, 브로드컴, 퀄컴 등이 되었다. 데이터 수집은 2000년 1월 1일부터 2018년 11월 중순까지의 데이터를 Yahoo API[16]를 통해 이루어졌다. 원시 데이터는 날짜, 시작가, 최고가, 최저가, 종가, 수정종가, 거래량으로 구성되어 있다.

모델 구동 시 입력된 파라미터는 두 모델 모두 같은 조건으로 학습률(Learning Rate)은 0.001, 훈련 배치 사이즈(Train Batch Size)와 검증 배치 사이즈(Validation Batch Size)는 기본 값으로 32, 이전 5개 데이터로 다음 데이터 1개를 예측하는 형태이므로 타입 스텝은 5, 에폭(Epoch) 횟수는 100으로 설정하였다.

5. 결론

과거 정보화 시대를 통해 산업 전반에 걸쳐 시스템 개발과 통신 인프라 구축에 힘입어 빅데이터가 축적되었고 이를 수용할 수 있는 하드웨어 자원 및 머신러닝 기술의 발전으로 고부가가치를 창출하기 위한 시도들이 다양하게 이루어지고 있다. 특히 복잡한 데이터에 대해서는 딥러닝 기술이 탁월한 효과를 보이고 있다. 주식과 같은 금융 분야에도 예외 없이 딥러닝 모델들을 적용하여 효과를 보고 있으며, LSTM 모델은 시계열 데이터에 대한 예측 모델로서 검증된 방법이다. 이에 본 연구에서는 LSTM 모델의 성능향상을 위해 고려해야 할 복잡한 매개변수 설정과 적용 함수들에 대해 실증적인 실험을 통해 적합한 방법을 제시하였다. 크게 가중치와 바이어스에 대한 초기화 대상과 방법들의 설정 방법, 과적합을 피하기 위해 정규화를 적용할 수 있는 대상과 방법, 활성화 함수의 종류와 적합한 방법, 최적화 알고리즘의 적용방법 등에 대해서 복합적인 형태로 실험을 수행하여 제시하였다. 물론 주가 데이터 셋의 대상과 경향패턴, 입력 데이터의 길이 등과 같은 데이터 모델링과 드롭아웃 비율, 학습률 설정 등의 하이퍼파라미터 설정 방법에 따라 학습 성능이 달라질 수도 있으나 이들을 다른 딥러닝 모델의 학습 환경에서 일반화하여 제시하는 설정값으로 고정한 상태에서 실험하였다. 다만 본 연구에서는 주가 예측에서 성능향상을 위해 중요하게 고려해야 할 복잡한 매개변수 항목들과 함수들은 어떤 것들이 있고 이들을 어떻게 설정해야 하는지에 대해 실험을 통해 제시함으로써 모델 적용 시 복잡성을 줄일 수 있게 하였다는 측면에서 의의가 있다고 생각된다. 향후에는 입력 데이터의 포맷과 길

이, 하이퍼파라미터들에 대한 성능평가를 추가적으로 수행함으로써 주요 설정 항목들의 조합에 대한 일반화 연구를 수행하고자 한다.

REFERENCES

[1] Alizadeh, M., et al. (2011). *An adaptive neuro fuzzy system for portfolio analysis*, *International Journal of Intelligent Systems, 22(2)*, 99-114.

[2] Behnoush Shakeri et al (2015). *Fuzzy Clustering Rule-Based Expert System for Stock Price Movement Prediction*, *NAFIPS, Redmond, Washington, USA, August 17-19*.

[3] R. Lakshman Naik & D. Ramesh & B. Manjula & Dr. A. Govardhan (2012). *Prediction of Stock Market Index Using Genetic Algorithm*, *Computer Engineering and Intelligent Systems, 3(7)*.[1]
H. J. Kim et al. (2018). Stock Price Prediction Using Deep Learning Ensemble, *SIGDB 34(2)*, 113-120.

[4] S. W. Kim & H. C Ahn (2010). Development of an Intelligent Trading System Using Support Vector Machines and Genetic Algorithms, *Journal of Intelligence and Information Systems, 16(1),* 71-92.

[5] J. Y. Heo & J. Y. Yang. (2015). SVM based Stock Price Forecasting Using Financial Statements, *KIISE transactons on computing practices, 21(3)*, 167-172.

[6] G. B. Nam et al. (2017). Development of Stock Investment System Using Machine Learning, *2017 Proceeding of Information Processing Society Fall Conference 24(2)*, 810-812.

[7] Mao, H. Zeng & X. J, Leng, & G, Zhai (2011). *Twitter mood predicts the stock market*, *Journal of Computational Science, 2(1)*, 1-8.

[8] Y. J. Song & J. W. Lee & J. W. Lee (2017). Performance Evaluation of Price-based Input Featuress in Stock Price Prediction using Tensorflow, *KIISE transactons on computing practices, 23(11)*.

[9] D. H. Shin, K. H. Choi & C. B. Kim. (2017). Deep Learning Model for Prediction Rate Improvement of Stock Price Using RNN and LSTM, *Jounral of KIIT, 15(10)*, 9-16.

[10] Ashwin Siripurapu (2015). *Convolutional Networks for stock Trading*, Stanford University.

[11] H. J. Kim et al. (2018). Stock Price Prediction Using Deep Learning Ensemble, *SIGDB 34(2)*, 113-120.

[12] W. S. Lee. (2017). A Deep Learning Analysis of the KOSPI's directions, Jounral of the Korean Data & Information Science Society, *28(2)*, 287-295.

[13] T. W. Kim & H. Y. Kim (2019). *Forecasting stock prices with a feature fusion LSTM-CNN model using different representations of the same data*, *PLoS ONE, 14(2)*.

[14] Klaus Greff et al. (2017). *LSTM: A Search Space Odyssey*, *IEEE Transactions on Neural Networks and Learning Systems, 28(10)*, 2222-2232.

[15] *Keras LSTM tutorial - How to easily build a powerful deep learning language mode*l, https://adventuresinmachinelearning/keras-lstmtutorial/

[16] Yahoo Finance. https://finance.yahoo.com6

정 종 진(Jongjin Jung) [정회원]

· 1992년 2월 : 인하대학교 전자계산공학과(공학사)
· 1995년 2월 : 인하대학교 전자계산공학과(공학석사)
· 2000년 2월 : 인하대학교 전자계산공학과(공학박사)
· 2002년 9월 ~ 현재 : 대진대학교 휴먼IT융합학부 교수
· 관심분야 : 빅데이터, 머신러닝, 딥러닝, 지식기반 시스템
· E-Mail : jjjung@daejin.ac.kr

김 지 연(Jiyeon Kim) [정회원]

· 1992년 2월 : 인하대학교 전자계산공학과(공학사)
· 1997년 2월 : 인하대학교 전자계산공학과(공학석사)
· 2008년 2월 : 인하대학교 전자계산공학과(공학박사)
· 2015년 3월 ~ 현재 : 대진대학교 창의미래인재대학 조교수
· 관심분야 : 빅데이터, 추천, 머신러닝, 딥러닝
· E-Mail : jykim629@daejin.ac.kr