

RNN모델에서 하이퍼파라미터 변화에 따른 정확도와 손실 성능 분석

김준용¹, 박구락^{2*}

¹서울신학대학교 IT융합소프트웨어학과 교수, ²공주대학교 컴퓨터공학부 교수

Analysis of Accuracy and Loss Performance According to Hyperparameter in RNN Model

Joon-Yong Kim¹, Koo-Rack Park^{2*}

¹Professor, Department of IT Convergence Software, Seoul Theological University

²Professor, Division of Computer Science & Engineering, Kongju National University

요약 본 논문은 감성 분석에 사용되는 RNN 모델의 최적화를 얻기 위한 성능분석을 위하여 하이퍼파라미터 튜닝에 따른 손실과 정확도의 추이를 관찰하여 모델과의 상관관계를 연구하였다. 연구 방법으로는 시퀀셜데이터를 처리하는데 가장 최적화된 LSTM과 Embedding layer로 히든레이어를 구성한 후, LSTM의 Unit과 Batch Size, Embedding Size를 튜닝하여 각각의 모델에 대한 손실과 정확도를 측정하였다. 측정 결과, 손실은 41.9%, 정확도는 11.4%의 차이를 나타내었고, 최적화 모델의 변화추이는 지속적으로 안정적인 그래프를 보여 하이퍼파라미터의 튜닝이 모델에 지대한 영향을 미침을 확인하였다. 또한 3가지 하이퍼파라미터 중 Embedding Size의 결정이 모델에 가장 큰 영향을 미침을 확인하였다. 향후 이 연구를 지속적으로 이어나가 모델이 최적의 하이퍼파라미터를 직접 찾아낼 수 있는 알고리즘에 대한 연구를 지속적으로 이어나갈 것이다.

주제어 : 순환신경망, 단순 순환신경망, 장.단기 기억레이어, 임베딩레이어, 하이퍼파라미터, 감성분석

Abstract In this paper, in order to obtain the optimization of the RNN model used for sentiment analysis, the correlation of each model was studied by observing the trend of loss and accuracy according to hyperparameter tuning. As a research method, after configuring the hidden layer with LSTM and the embedding layer that are most optimized to process sequential data, the loss and accuracy of each model were measured by tuning the unit, batch-size, and embedding size of the LSTM. As a result of the measurement, the loss was 41.9% and the accuracy was 11.4%, and the trend of the optimization model showed a consistently stable graph, confirming that the tuning of the hyperparameter had a profound effect on the model. In addition, it was confirmed that the decision of the embedding size among the three hyperparameters had the greatest influence on the model. In the future, this research will be continued, and research on an algorithm that allows the model to directly find the optimal hyperparameter will continue.

Key Words : RNN, SimpleRNN, LSTM Layer, Embedding Layer, Hyper-parameter, Sentiment Analysis

1. 서론

최근 인공지능 모델을 이용한 다양한 서비스가 우리 주위에 넘쳐나고 있다. 자동화 시스템에 인공지능을 적용한 현대의 오토마타(Automata)는 마치 인간의 지능

을 가진 것과 같이 스스로 생각하고 행동하는 것처럼 느껴지게 한다.

인공지능은 각각의 성격과 데이터 처리의 특성에 따라 텍스트 분석과 이미지 분석에 적합한 DNN(Deep Neural

*This article is extended and excerpted from the conference paper presented at ICSMB2021

*Corresponding Author : Koo-Rack Park(ecgrpark@kongju.ac.kr)

Network), 이미지 분류에 사용되는 CNN(Convolutional Neural Network)와 감성 분석과 시퀀셜 데이터를 다루는 RNN(Recurrent Neural Network)로 대별되며, 특히 언어와 같은 시퀀셜 구조의 데이터를 대상으로 하는 인공지능 모델은 RNN구조가 매우 적합하다[1].

이러한 RNN모델의 성능을 측정하기 위한 지표로 모델의 정확도를 이용하여 검증하는 연구 결과가 자연어처리 모델 성능 검증과 전력에너지의 시계열 예측을 위한 연구 등 다양한 분야에서 활용되고 있으며[2, 3], 모델의 최적화를 얻기 위한 전제조건인 모델의 정확도와 이를 결정하는 손실률을 최소화는 각종 파라미터의 최적화 연구가 필수적으로 이루어지고 있다[4, 5].

그러나 RNN모델의 정확도, 손실률과 하이퍼파라미터들 사이의 연관성에 대한 연구는 상대적으로 부족한 것이 현실이다.

본 논문에서는 RNN모델의 성능을 결정하는 하이퍼파라미터 중 가장 비중이 큰 Unit, Batch Size와 Embedding Size의 값을 튜닝하여 하이퍼파라미터 변화에 따른 모델의 최적화를 측정하였고, 각각의 파라미터가 모델에 미치는 변화를 연구하였다.

실험 결과를 모니터링하여 하이퍼파라미터에 따른 모델의 상관관계를 확인한 결과, 최적화된 모델의 정확도와 손실은 연속성을 가진 그래프로 나타나며, 일정 기점을 통해 서서히 하강함을 볼 수 있었다. 이러한 결과는 하이퍼파라미터의 조합에 따라 모델의 최적화가 이루어짐을 알 수 있었고, 이것이 모델을 최적화 하는데 중요한 요소 중의 하나임을 증명하였다.

특히, 모델의 최적화에 직접적인 영향을 미치는 Unit과 Batch Size 그리고 Embedding Size 중 Embedding Size의 크기가 가장 큰 영향을 미침을 확인하였다. 이러한 결과는 RNN모델을 설계 시 연속성 있는 Embedding Size에 대한 충분한 고려가 반드시 필요하다는 것을 확인하였다.

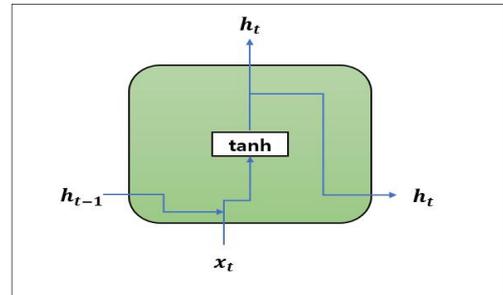
본 논문의 구성은 제 2 장에서는 관련연구를 통해 본 논문에서 연구하고자 하는 RNN모델, LSTM, Embedding, RNN과 하이퍼파라미터, 그리고 하이퍼파라미터를 튜닝하여 RNN모델의 최적화를 연구한 기존연구에 대하여 설명하였다. 제 3 장에서는 본 논문에서 연구한 하이퍼파라미터 튜닝에 의한 RNN모델의 성능분석을 위한 실험 및 고찰을 기술하였고, 마지막으로 제 4 장을 통하여 실험에서 얻은 결과의 의미와 향후 연구방향을 제시하였다.

2. 관련연구

2.1 SimpleRNN

순환 신경망의 기본 레이어는 SimpleRNN으로 RNN구조의 가장 간단한 구조로 구성되어있다. 순환 신경망은 시간의 흐름에 따른 입력값의 변화를 반영함으로써 시계열 형태의 데이터를 처리하는데 주로 사용되고 있다[6]. 합성곱 필터를 사용하여 이미지를 판단하는 CNN모델과 달리 RNN은 각각의 유닛이 자신의 출력 값을 가중치를 반영하여 다음 유닛에게 입력 값으로 전달함으로써 순차적으로 처리되어야 하는 시계열 데이터를 처리한다[7].

다음의 Fig. 1은 SimpleRNN의 기본적인 구조도이다[8].



h_{t-1} : output of previous timestep
 x_t : input
 h_t : output
 $h_t = \tanh(W_{hh}h_{t-1} + W_{hx}x_t + b_n)$

Fig. 1. SimpleRNN Layer

모델은 이전 타임스텝에서 출력으로 나온 결과와 현재의 입력값을 tanh함수를 통해 활성화하여 타임스텝 방향과 출력방향으로 데이터를 전송하는 셀들로 시퀀셜하게 구성되어있다.

이와 같은 SimpleRNN 레이어는 간단한 형태로 빠르게 모델을 학습시키고 활용하는데 사용된다.

2.2 LSTM

LSTM(Long-Short Term Memory Network)는 SimpleRNN사용시 발생하는 입력 데이터의 증가에 따른 타임스텝 증대로 인한 모델의 학습능력 저하 문제를 해결한 것으로 다음의 Fig. 2와 같은 구조를 가지고 있다[8].

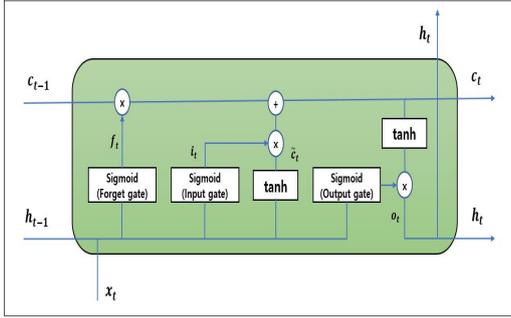


Fig. 2. LSTM Layer

LSTM 레이어는 시스모이드 함수의 특성을 이용하여 필요한 정보만 통과시킨다. 시그모이드 함수는 Forget, Input, Output 3개의 gate로 구성되며 다음의 수식과 같이 데이터를 처리하도록 모델링한다.

첫째, 입력과 출력에 가중치를 적용하여 3개의 gate를 통과시켜 출력(3)을 구한다.

$$i_t = \text{sigmoid}(W_i h_{t-1} + U_i x_t + b_i) \quad (1)$$

$$f_t = \text{sigmoid}(W_f h_{t-1} + U_f x_t + b_f) \quad (2)$$

$$o_t = \text{sigmoid}(W_o h_{t-1} + U_o x_t + b_o) \quad (3)$$

둘째, simpleRNN방식과 동일하게 입력 값인 x_t 과 이전 타임스텝의 출력인 h_{t-1} 에 각각의 가중치를 적용하여 셀 상태가 되기 전 값을 구한다.

$$\tilde{c}_t = \tanh(W_c h_{t-1} + U_c x_t + b_c) \quad (4)$$

셋째, 이전 타임스텝의 셀 상태 반영도를 결정하고 새로 입력된 출력과 이를 곱한 값을 합하여 다음 타임스텝의 셀을 형성한다.

$$c_t = f_t \times c_{t-1} + i_t \times \tilde{c}_t \quad (5)$$

마지막으로 활성화 함수를 Output gate출력 값에 곱하여 출력값을 구한다[9-11].

$$h_t = \tanh(c_t) \times o_t \quad (4)$$

위의 과정을 통하여 타임스텝 구간을 연속적으로 끊기지않고 통과하는 셀을 형성하여 장기 의존성으로 문제가 되는 simpleRNN을 대체하여 사용되고 있다. 본

연구에서는 이러한 LSTM의 특성을 고려하여 연구모델에 LSTM을 적용하였다.

2.3 Embedding

임베딩 레이어는 자연어를 수치로 바꿔 정보를 처리하기 위한 은닉층으로 주로 사용된다[12].

CNN에서 이미지를 픽셀 단위로 소분하여 사용하듯이 RNN은 자연어를 문자 단위로 나누어 사용할 수 있다. 자연어 처리 기법은 n-gram, 3-gram방식으로 이루어졌으나 많은 양의 데이터를 쉽게 처리할 수 있는 인덱스 방식의 원-핫 인코딩 방식이 주로 많이 사용되었다. 사용하는 메모리의 양에 비해 처리되는 정보의 양이 적은 점과 저장된 단어가 많아질수록 원-핫 인코딩 배열을 두 번째 차원 크기가 늘어나 자원을 많이 소모하는 단점이 있다. 이러한 단점을 보완하여 디자인된 것이 임베딩 레이어로 자연어 처리에 가장 많이 사용되고 있다. 임베딩 레이어를 훈련시키는 방법으로는 Word2Vec, ELMo, FastText, GloVe 등이 주류를 이룬다[13-16].

2.3 RNN과 하이퍼파라미터

RNN모델의 성능을 결정하는 하이퍼파라미터는 다음의 Fig. 3과 같은 구조로 이루어져있다.

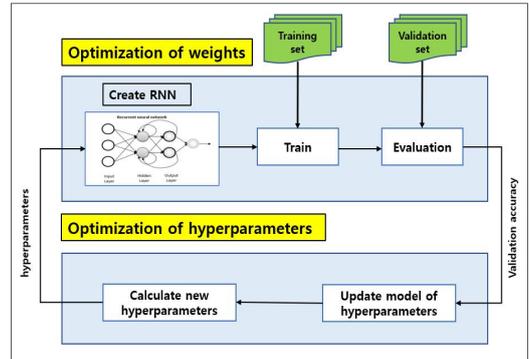


Fig. 3. Structure of RNN on Hyperparameters

하이퍼파라미터는 데이터의 훈련과 검증을 통해 형성된 초기모델을 최적화하기 위하여 새로운 조합의 하이퍼파라미터를 생성한다. 생성된 하이퍼파라미터를 이용하여 새로운 RNN모델을 형성하고 이를 다시 훈련시켜 검증하는 과정을 반복하여 최적의 모델을 찾는 구조를 가지고 있다.

또한, 본 논문의 연구와 관련된 기존 관련연구로 “한

국어에 적합한 단어 임베딩 모델 및 파라미터 튜닝에 관한 연구”가 있었으며, 연구 결과 하이퍼파라미터 튜닝을 통해 Embedding Size가 300일 때 최적의 모델을 구하는 연구가 선행되었다[17].

3. 하이퍼파라미터 튜닝에 따른 정확도와 손실을

실험 및 고찰

본 연구는 연구에 필요한 데이터 셋을 구축하고 이를 학습시키는 인공지능 모델을 다양한 하이퍼 파라미터 별로 테스트하여 각각의 정확성과 손실을 측정 후, 이들의 상관관계를 비교 확인하여 변동추이와 경향성 등을 비교하여 하이퍼파라미터의 적용에 따른 모델의 최적화 변화와 하이퍼파라미터들간의 연관성을 연구하였다.

본 연구를 진행하기 위한 실험환경의 프레임워크는 텐서플로우 2.0을 사용하였고, 순환된 출력층의 연결을 위하여 Fatten층과 Dense층을 추가하였으며, 활성화 함수는 시그모이드 함수를 이용하였다. 이렇게 구성된 모델을 최적화하기 위한 최적화 함수는 Adam을 적용하여 실험을 진행하였다.

하이퍼파라미터 변화에 따라 각각의 조합당 50회씩 총 1,500회의 실험을 통해 얻은 결과를 표준편차를 이용하여 범주에서 벗어나는 10%의 데이터를 제외한 값들의 평균으로 손실률과 정확도를 구하였다.

본 연구에서 적용한 실험에 대한 모델의 구조와 실험과정은 다음과 같다.

다음의 Fig. 4는 본 연구에서 사용한 RNN모델의 구조도이다.

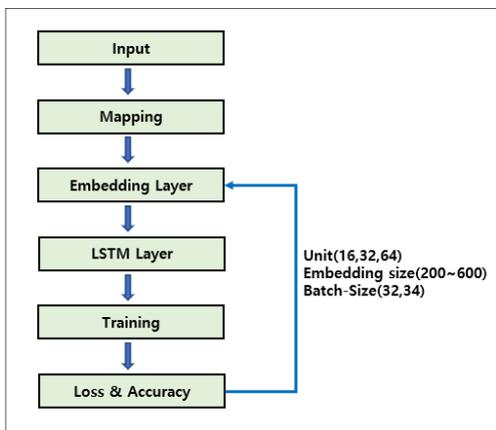


Fig. 4. Structure of Model

첫째, 데이터 셋은 RNN감성처리 기법을 적용할 수 있는 텍스트 형태의 데이터를 취하였다. 신뢰성 확보를 위하여 전 세계에서 가장 많이 판매되고 읽힌 책 중의 하나인 성경을 택하여 이 중 신약 부분에 예수의 직접적인 교훈이 기록되어 있는 공관 복음서를 데이터로 선택하였다. 이 중 일반적인 상용구를 제외한 의미 있는 문장들만을 택하여 이진분류를 이용하여 예수의 교훈과 아닌 것으로 분류하여 총 3,752문장으로 구성된 데이터 셋을 구축하였다.

다음의 Fig. 5와 6은 데이터 셋에 구축된 데이터의 처음과 끝부분이다.

No	Document	Label
0 400101	A record of the genealogy of Jesus Christ the ...	0
1 400102	Abraham was the father of Isaac, Isaac the fat...	0
2 400103	Judah the father of Perez and Zerah, whose mot...	0
3 400104	Ram the father of Amminadab, Amminadab the fat...	0
4 400105	Salmon the father of Boaz, whose mother was Ra...	0
5 400106	and Jesse the father of King David. David was ...	0
6 400107	Solomon the father of Rehoboam, Rehoboam the f...	0
7 400108	Asa the father of Jehoshaphat, Jehoshaphat the...	0
8 400109	Uzziah the father of Jotham, Jotham the father...	0
9 400110	Hezekiah the father of Manasseh, Manasseh the ...	0

Fig. 5. Data set(header)

No	Document	Label
3742 432116	Again Jesus said, "Simon son of John, do you t...	1
3743 432117	The third time he said to him, "Simon son of J...	1
3744 432118	Jesus said, "Feed my?sheep. I tell you the tru...	1
3745 432119	Jesus said this to indicate the kind of death ...	0
3746 432120	Peter turned and saw that the disciple whom Je...	0
3747 432121	When Peter saw him, he asked, "Lord, what abou...	0
3748 432122	Jesus answered, "If I want him to remain alive...	1
3749 432123	Because of this, the rumor spread among the?br...	0
3750 432124	This is the disciple who testifies to these th...	0
3751 432125	Jesus did many other things as well. If every ...	0

Fig. 6. Data set/footer)

둘째, 구축된 데이터를 학습시키고 학습된 데이터를 테스트하기 위하여 전체의 75%인 2,814문장을 학습데이터로 나누고, 나머지 25%인 938개의 문장은 테스트를 위하여 구분하였다.

셋째, RNN의 히든레이어는 입력 데이터의 길이에 비교적 영향을 적게 받는 LSTM과 Embedding으로 구성하여 모델을 구성하였다. 다음의 Fig. 7은 구성한 RNN모델을 나타낸 것이다.

```

model.add(layers.Flatten())
model.add(layers.Dense(1, activation='sigmoid'))
model.summary()

Model: "sequential_1"
-----
Layer (type)                Output Shape         Param #
-----
embedding_1 (Embedding)     (None, None, 400)   1467600
-----
lstm_2 (LSTM)               (None, 16)          26688
-----
flatten (Flatten)           (None, 16)           0
-----
dense (Dense)               (None, 1)            17
-----
Total params: 1,494,305
Trainable params: 1,494,305
Non-trainable params: 0
    
```

Fig. 7. RNN Model

넷째, 위에서 구성한 모델을 Unit과 Batch Size, Embedding Size를 변화시켜가면서 각각의 손실과 정확도를 확인한 결과 아래의 Table 1과 같은 결과를 얻었다.

Table 1. Change according to hyper-parameter

Unit	Batch Size	Embedding Size	Loss	Accuracy
16	32	200	0.6371	0.7745
		300	0.6939	0.7654
		400	0.6398	0.7923
		500	0.6803	0.7836
		600	0.6645	0.7973
	64	200	0.6721	0.7813
		300	0.5977	0.7904
		400	0.5079	0.8273
		500	0.5934	0.7996
		600	0.6328	0.7859
32	32	200	0.6103	0.7585
		300	0.6216	0.7904
		400	0.6718	0.8018
		500	0.8087	0.7676
		600	0.6441	0.7904
	64	200	0.6117	0.7881
		300	0.6020	0.7767
		400	0.6899	0.7790
		500	0.6467	0.7425
		600	0.6492	0.7972
64	32	200	0.6598	0.7835
		300	0.5837	0.7904
		400	0.6057	0.8018
		500	0.6220	0.7927
		600	0.6541	0.7972
	64	200	0.6206	0.7835
		300	0.6155	0.8018
		400	0.6205	0.7882
		500	0.8753	0.7722
		600	0.7435	0.7836

위의 Table 1에서 보이는 바와 같이 하이퍼파라미

터의 변화에 따라 손실과 정확도가 변화하는 것을 알 수 있다. 실험 데이터로 확인한 결과는 Unit이 16, Batch Size가 64 그리고 Embedding Size가 400일 때 최적화가 이루어졌음을 알 수 있다.

모델의 학습도와 검증을 위한 손실과 정확도의 수렴은 다음의 Fig. 8, 9와 같이 확인되었다.

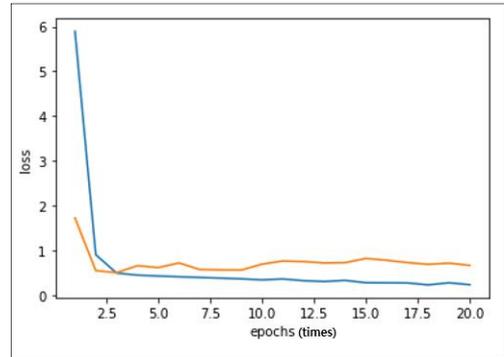


Fig. 8. Loss of model

Fig. 8에서 볼 수 있듯이 파란색으로 표현된 학습 데이터와 테스트 데이터의 손실이 유사한 경향을 보이며 유의미하게 수렴하고 있음을 알 수 있다.

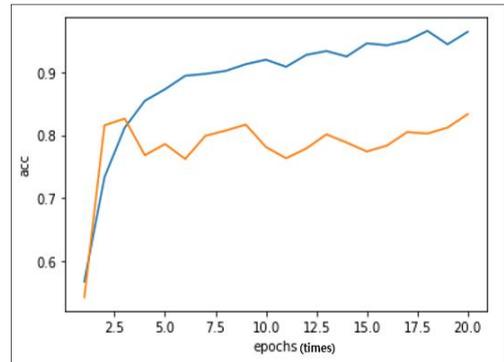


Fig. 9. Accuracy of model

Fig. 9의 경향을 볼 때 지속적으로 학습 데이터와 테스트 데이터의 정확도가 상승하고 있는 형태를 보이고 있어 본 연구의 데이터와 모델의 신뢰성을 보여주고 있다.

다섯째, 위에서 확인된 하이퍼파라미터의 변화에 따른 모델의 손실과 정확도의 상관관계는 다음의 Fig. 10, 11과 같다.

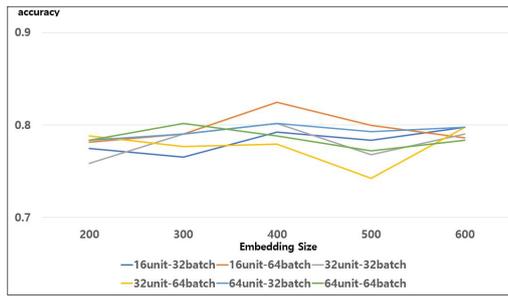


Fig. 10. Accuration comparison table

위의 Fig. 10에서 각각의 하이퍼파라미터 튜닝에 따른 정확도 변화 추이는 주황색으로 나타난 그래프가 가장 일정한 패턴을 보이며 Embedding Size가 400까지는 정확도가 상승하다가 이 지점을 기점으로 하락하는 추이를 보이고 있다.

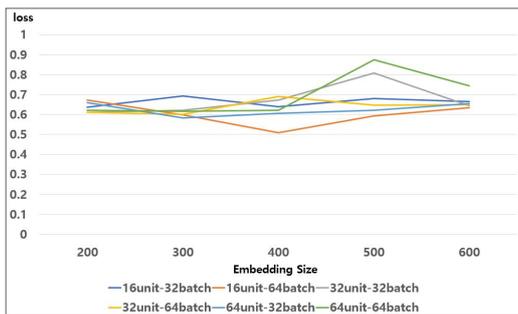


Fig. 11. Loss comparison table

위의 Fig. 11은 손실 추이를 확인하기 위한 그래프로 정확도 비교데이터와 같이 주황색 그래프가 일정한 패턴을 보이고 있다. 손실도 정확도와 같이 Embedding Size 400을 기점으로 지속적으로 낮아지다가 다시 상승하는 추이를 보이고 있다.

또한, 하이퍼파라미터들간의 연관관계를 확인한 구간 별 최적의 손실과 정확도는 다음의 Table 2-4와 같이 나타남을 확인하였다.

Table 2. Loss and Accuracy per Unit

Unit	Loss	Accuracy
16	0.5079	0.8273
34	0.6020	0.8018
64	0.5837	0.8018

Table 2에서 나타난 것 같이 Unit의 개수에 따른 각 단계별 최소 손실률과 최대 정확도는 Unit의 개수가 16개일 때 가장 최적화된 값을 나타내고 있다. 이는 Unit의 갯수가 증가할수록 손실과 정확도의 신뢰성이 떨어짐을 나타내며 본 모델에 최적화된 Unit의 수는 16개임을 알 수 있다.

다음의 Table 3과 Table 4는 Batch Size와 Embedding Size변화에 따른 손실과 정확도를 나타낸 것이다.

Table 3. Loss and Accuracy per Batch Size

Batch Size	Unit	Loss	Unit	Accuracy	Re mark
32	16	0.6371	16	0.7973	o
	32	0.6103	32	0.8081	o
	64	0.5837	64	0.8018	o
64	16	0.5079	16	0.8273	o
	32	0.6020	32	0.7972	o
	64	0.6205	64	0.8018	o

Table 3에서 나타난 바와 같이 Batch Size는 최적의 손실과 정확도를 나타내는 구간이 전 단계인 16개의 Unit을 가진 구간에서 동일하게 나타나며 다른 구간 역시 동일한 Unit구간에 있음을 알 수 있다. 이는 Unit의 영향에 매우 종속적임을 나타낸다.

다음의 Table 4는 Embedding Size 튜닝에 따른 손실과 정확도의 변화를 확인한 결과이다.

Table 4. Loss and Accuracy per Embedding Size

Embedding Size	Unit-Batch	Loss	Unit-Batch	Accuracy	Re mark
200	16-32	0.6103	32-64	0.7881	x
300	64-32	0.5837	64-64	0.8018	x
400	16-64	0.5079	16-64	0.8273	o
500	16-64	0.5934	16-64	0.7996	o
600	16-64	0.6328	32-64 64-32	0.7972	x

Table 4에서 나타난 바와 같이 최적화를 나타내는 구간에서의 하이퍼파라미터는 모델의 전체적인 성능을 따르나, 각각의 구간별 최적화 값은 전 단계인 Unit과 Batch Size에 영향을 받지 않는 구간이 발생함을 알 수

있다. 이는 모델의 최적화를 위해 Embedding Size의 크기 결정에 대한 면밀한 검토가 필요함을 의미한다.

실험 결과를 종합해볼 때 하이퍼파라미터의 조합에 따라 RNN모델의 최적화에 유의미한 영향을 미침을 알 수 있다. 특히, Unit과 Batch Size보다는 Embedding Size가 모델의 최적화에 미치는 영향이 크음을 확인할 수 있다.

4. 결론

본 연구는 RNN모델의 성능분석을 정확도 중심으로 연구한 기존의 연구를 바탕으로 성능분석을 위한 모델의 정확도와 손실에 직접적 영향을 미치는 하이퍼파라미터들이 RNN모델에 어떠한 영향을 미치는지를 확인하였다.

실험을 통해 RNN모델의 하이퍼파라미터를 변화시켜가며 손실과 정확도를 측정하고 하이퍼파라미터의 변화가 모델의 최적화에 영향을 미치는 것을 확인하였다. 손실은 최소와 최대값이 0.5079와 0.8753을 기록하여 41.9%의 차이를 보였고, 정확도 또한 0.8273과 0.7425를 나타내 11.4%의 차이를 나타냈다.

그러나 이러한 정량적 수치보다 하이퍼파라미터 변화에 따른 손실과 정확도의 추이를 살펴보면 최적화 모델은 지속적인 추이를 보였지만 다른 모델들은 이산적인 추이를 보여 안전성에도 문제가 있다는 것을 확인할 수 있다.

또한 Unit의 개수와 Batch Size의 결정은 매우 밀접한 상호 의존적 경향성을 가지고 있으나 Embedding Size는 전 단계 하이퍼파라미터들과의 연관성이 낮은 것으로 확인되었다.

이는 RNN모델의 최적화를 위한 연산에서는 Unit의 갯수는 가능한 작은 것이 유리하며, Batch Size는 한번에 처리하는 데이터의 크기이므로 컴퓨테이션이 허락하는 한도내에서는 클수록 효과적이므로 이러한 영향에 의해 실험모델의 Unit과 Batch Size가 16-64일 때 최적화를 나타냈다. Embedding size는 실험 결과에서 보여지듯 뚜렷한 경향성을 나타내고 있다. 일정한 패턴으로 상승하다가 정점을 지나 하강하는 그래디언트적 경향을 보이므로서 최적의 모델을 찾기 위해서는 반드시 Embedding Size에 대한 연속적인 탐색이 고려되어야 한다.

향후 이러한 하이퍼파라미터의 최적화를 모델 내에서 자체적으로 찾아내어 안전성과 효율을 최대화할 수

있는 알고리즘에 대한 연구가 지속적으로 이루어져야 할 것이다.

REFERENCES

- [1] G. Y. Lee & S. B. Lee. (2018). Universal Prediction System Realization Using RNN. *The Journal of Korean Institute of Information Technology*, 16(10), 11-20.
DOI : 10.14801/jkiit.2018.16.10.11
- [2] C. Y. Lee & T. G. Lee, Kyungseop Shin. (2019). Performance Comparison of Natural Language Processing Model Based on Deep Neural Networks. *The Journal of Korean Institute Communications and Information Sciences*, 44(7), 1344-1350
- [3] C. Y. Lee & J. Kim, (2018). The prediction and analysis of the power energy time series by using the elman recurrent neural network. *Journal of the Society of Korea Industrial and Systems Engineering*, 41(1), 84-93.
DOI : 10.11627/jkise.2018.41.1.084
- [4] Y. Kim et al. (2018). Performance evaluation of machine learning and deep learning algorithms in crop classification: Impact of hyperparameters and training sample size. *Korean Journal of Remote Sensing*, 34(5), 811-827.
DOI : 10.7780/kjrs.2018.34.5.9
- [5] I. Aliyu, R. M. Mahmood & C. G. Lim. (2019). LSTM Hyperparameter Optimization for an EEG-Based Efficient Emotion Classification in BCI. *The Journal of the Korea institute of electronic communication sciences*, 14(6), 1171-1180.
DOI : 10.37727/jkdas.2019.21.4.1771
- [6] S. Karita et al. (2019, December). A comparative study on transformer vs rnn in speech applications. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)* (pp. 449-456). IEEE.
DOI : 10.1109/ASRU46091.2019.9003750
- [7] K. Rao, H. Sak & R. Prabhavalkar. (2017, December). Exploring architectures, data and units for streaming end-to-end speech recognition with rnn-transducer. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)* (pp. 193-199). IEEE.
DOI : 10.1109/ASRU.2017.8268935
- [8] Colah' Blog. (2015). *Understanding LSTM Networks*.

(Online). <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>

- [9] X. Zhang, F. Chen & R. Huang. (2018). A combination of RNN and CNN for attention-based relation classification. *Procedia computer science*, 131, 911-917.
- [10] C. Baziotis, N. Pelekis & C. Doukeridis. (2017, August). Datastories at semeval-2017 task 6: Siamese LSTM with attention for humorous text comparison. *In Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)* (pp. 390-395).
- [11] D. A. Reddy, M. A. Kumar & K. P. Soman. (2019). *c In Soft Computing and Signal Processing* (pp. 385-394). Springer, Singapore.
- [12] Z. Li, R. Kulhanek, S. Wang, Y. Zhao & S. Wu. (2018, April). Slim embedding layers for recurrent neural language models. *In Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 32, No. 1).
DOI : 10.1007/978-981-13-3393-4_40
- [13] T. Mikolov, I. Sutskever, K. Chen, G. Corrado & J. Dean. (2013). Distributed representations of words and phrases and their compositionality. *arXiv preprint arXiv:1310.4546*.
- [14] N. Reimers & I. Gurevych. (2019). Alternative weighting schemes for elmo embeddings. *arXiv preprint arXiv:1904.02954*.
- [15] I. Santos, N. Nedjah & L. de Macedo Mourelle. (2017, November). Sentiment analysis using convolutional neural network with fastText embeddings. *In 2017 IEEE Latin American Conference on Computational Intelligence (LA-CCI)* (pp. 1-5). IEEE.
DOI : 10.1109/LA-CCI.2017.8285683
- [16] A. Tifrea, G.Bécigneul & O. E. Ganea. (2018). Poincaré GloVe: Hyperbolic Word Embeddings. *arXiv preprint arXiv:1810.06546*.
- [17] S. Choi, J. Seol & S. G. Lee. (2016), *On Word Embedding Models and Parameters Optimized for Korean*. Korean Language Information Science Society, 252-256.

김 준 용(Joon-Yong Kim)

[정회원]



- 1985년 2월 : 성균관대학교 토목공학과(학사)
- 2013년 2월 : 공주대학교 컴퓨터공학과(석사)
- 2015년 3월 : 공주대학교 컴퓨터공학과(박사)

- 2015년 3월 ~ 2020 2월 : 공주대학교 컴퓨터공학과 외래교수
- 2018년 3월 ~ 2020년 2월 : 배재대학교 게임공학과 외래교수
- 2017년 3월 ~ 2020년 2월 : 대전보건대학교 바이오정보과 겸임교수
- 2020년 3월 ~ 현재 : 서울신학대학교 IT융합소프트웨어학과 조교수
- 관심분야 : Machine Learning, AutoML, Artificial Intelligence
- E-Mail : musimk@stu.ac.kr

박 구 락(Koo-Rack Park)

[정회원]



- 1986년 2월 : 중앙대학교 전기공학과(공학사) 부전공(전자계산)
- 1988년 2월 : 숭실대학교 전자계산학과(공학석사)
- 2000년 2월 : 경기대학교 전자계산학과(이학박사)

- 1991년 4월 ~ 현재 : 공주대학교 컴퓨터공학부 교수
- 관심분야 : IT 컨버전스, 인공지능, 빅데이터, 영상처리
- E-Mail : ecgrrpark@kongju.ac.kr