KIPS Trans. Softw. and Data Eng.
Vol.11, No.6 pp.245~254
ISSN: 2287-5905 (Print), ISSN: 2734-0503 (Online)

https://doi.org/10.3745/KTSDE.2022.11.6.245

Comparison of Deep Learning Models Using Protein Sequence Data

Jeung Min Lee† · Hyun Lee††

ABSTRACT

Proteins are the basic unit of all life activities, and understanding them is essential for studying life phenomena. Since the emergence of the machine learning methodology using artificial neural networks, many researchers have tried to predict the function of proteins using only protein sequences. Many combinations of deep learning models have been reported to academia, but the methods are different and there is no formal methodology, and they are tailored to different data, so there has never been a direct comparative analysis of which algorithms are more suitable for handling protein data. In this paper, the single model performance of each algorithm was compared and evaluated based on accuracy and speed by applying the same data to CNN, LSTM, and GRU models, which are the most frequently used representative algorithms in the convergence research field of predicting protein functions, and the final evaluation scale is presented as Micro Precision, Recall, and F1-score. The combined models CNN-LSTM and CNN-GRU models also were evaluated in the same way. Through this study, it was confirmed that the performance of LSTM as a single model is good in simple classification problems, overlapping CNN was suitable as a single model in complex classification problems, and the CNN-LSTM was relatively better as a combination model.

Keywords : CNN, LSTM, GRU, Combined Model, Protein Sequence

단백질 기능 예측 모델의 주요 딥러닝 모델 비교 실험

이 정 민† · 이 현††

요 약

단백질은 모든 생명 활동의 기본 단위이며, 이를 이해하는 것은 생명 현상을 연구하는 데 필수적이다. 인공신경망을 이용한 기계학습 방법론이 대두된 이후로 많은 연구자들이 단백질 서열만을 사용하여 단백질의 기능을 예측하고자 하였다. 많은 조합의 딥러닝 모델이 학계에 보고되었으나 그 방법은 제각각이며 정형화된 방법론이 없고, 각기 다른 데이터에 맞춰져있어 어떤 알고리즘이 더 단백질 데이터를 다루는 데 적합한지 직접 비교분석 된 적이 없다. 본 논문에서는 단백질의 기능을 예측하는 융합 분야에서 가장 많이 사용되는 대표 알고리즘인 CNN, LSTM, GRU 모델과 이를 이용한 두가지 결합 모델에 동일 데이터를 적용하여 각 알고리즘의 단일 모델 성능과 결합 모델의 성능을 정확도와 속도를 기준으로 비교 평가하였으며 최종 평가 척도를 마이크로 정밀도, 재현율, F1 점수로 나타내었다. 본 연구를 통해 단순 분류 문제에서 단일 모델로 LSTM의 성능이 준수하고, 복잡한 분류 문제에서는 단일 모델로 중첩 CNN이 더 적합하며, 결합 모델로 CNN-LSTM의 연계 모델이 상대적으로 더 우수함을 확인하였다.

키워드 : CNN, LSTM, GRU, 결합 모델, 단백질 서열

1. 서 론

단백질은 약 20종류의 아미노산이 사슬처럼 이어져 만들어지는 생체 고분자의 일종이다. 단일 또는 복합으로 다른 단백질이나 분자와 상호작용하여 여러 가지 생명 활동을 매개하거나 수행한다. 대부분의 세포 기능에 관여하며 제각기 기능에 따라 다른 형태와 다른 물성을 지닌다. 생체 내 중요한 역할을 담당하는 효소와 호르몬, 항체 또한 단백질의 일종이다. 따라서 단백질은 모든 생명 활동의 기본 단위이며, 이를 이해하는 것은 생명 현상을 연구하는 데 필수적이다.

단백질은 DNA에 담겨 있는 생명 활동의 정보가 체내에서 단백질로 해독될 때 정해진 아미노산 서열에 의해 그 구조와 기능이 결정된다. 이에 많은 연구자가 단백질 서열정보만 이용하여 단백질의 기능과 구조를 예측하고자 하였고, 인공신경망을 이용한 기계학습 방법론이 대두된 이후로 해당 분야는 더욱 활발하게 연구되고 있다. ECPred(2018), EnzyNet(2018),

※ 본 논문은 교육부 및 한국연구재단의 4단계 두뇌한국21 사업(4단계 BK21 사업)으로 지원된 연구임.
† 준 회 원 : 선문대학교 컴퓨터융합전자공학과 바이오빅데이터융합 석사과정
†† 정 회 원 : 선문대학교 컴퓨터공학부 부교수
Manuscript Received : December 10, 2021
First Revision : January 6, 2022
Second Revision : February 9, 2022
Accepted : February 22, 2022
* Corresponding Author : Hyun Lee(mahyun91@sunmoon.ac.kr)

※ This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (http://creativecommons.org/ licenses/by-nc/3.0/) which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

MF-EFP(2020), UDSMProt(2020), DeEPn (2020), DeepEC (2020), HECNet (2020)과 같이 Enzyme commission number (EC number)를 통해 효소의 기능을 예측하는 연구[1-7], rawMSA(2019), DeepACLSTM(2019), E. C. Alley et al(2019)와 같은 단백질 2차 구조를 예측하는 연구[8-10], DeepDom(2019)과 같이 domain 영역을 예측하는 연구[11], Min et al (2021)과 같이 열충격 단백질을 예측하는 연구[12], PRIAM(2018), mApLe(2020)와 같이 생물의 대사 경로를 예측하는 연구[13,14], ET-GRU(2019)와 같이 전사 지점을 예측하는 연구[15] 등, 아미노산 서열정보를 이용한 단백질의 기능 및 구조 예측 연구는 이토록 단백질이 관여하는 기능만큼이나 무척 다양하고 광범위하다.

그러나 필드가 넓은 만큼 같은 연구 분야라고 하더라도 같은 알고리즘을 적용하는 방식은 전부 제각각이다. 이미지 프로세싱이나 객체 검출 분야의 연구와는 달리 문자열 데이터를 전처리하기 위한 임베딩(Embedding) 방식부터 훈련 방식에 이르기까지 명확히 확인되거나 보편적으로 정해진 기준이 없고 비슷한 결과물을 도출하는 개별 모델에 대한 상대적인 평가는 존재하나 어떤 알고리즘이 아미노산 서열정보를 다루는데 더 적합한지 직접 비교 분석된 적은 없다. Z. Tao (2020)에서는 각 연구 모델마다 지원하는 데이터, 목표로 삼는 데이터가 다 다르므로 현존하는 모델을 다른 모델과 비교하는 것은 의미 없다고 언급한 바 있으나[16] 그것은 이미 특정 데이터에 맞춰 학습이 완전히 끝난 모델의 실험 결과끼리 비교했을 때이며 알고리즘 자체나 알고리즘 적용법 또는 방법론에 대한 비교가 아니다. 따라서 본 논문에서는 2020년 기준 단백질 서열을 이용한 기능과 구조 예측 분야에서 가장 많이 사용되고 있는 딥러닝 모델인 CNN과 LSTM/GRU 모델을 살펴보고, 동일한 데이터 전처리 임베딩 방식을 사용했을 때 각 모델이 보이는 단일 성능과 CNN+RNN 계열 결합 모델 두 가지의 성능을 비교 평가하고자 한다.

공정한 평가를 위해 모든 알고리즘은 선행 연구에서 사용된 방법론을 따르되 결과물은 전부 효소번호(Enzyme Commission Number; EC number) 예측기로 구성되었다. 효소 번호는 단백질의 일종인 효소의 기능을 4자리 숫자로 정리하여 표현한 것이다. 각 모델 간의 비교 평가 기준은 정확도(Accuracy)와 손실(Loss), 학습 속도, 테스트 속도이고 최종 결과는 마이크로 정밀도, 재현율, F1 점수(Micro Recall, Precision, F1-score)로 수치화하였다.

본 논문의 구성은 다음과 같다. 2장에서 효소 번호와 기능의 표기법, 합성곱 신경망(CNN), 순환 신경망(RNN) 알고리즘 계열인 LSTM과 GRU 알고리즘을 살펴본다. 3장에서 실험 방법을 소개한다. 4장에서 실험 결과를 분석한다. 속도와 정확도를 기준으로 각 알고리즘을 평가하고 비교한 뒤, CNN+RNN 계열의 결합 모델 두 가지를 같은 기준으로 평가한다. 5장에서 결론을 짓고 논문을 마무리한다.

2. 관련 연구

2.1 효소 번호

효소란 특수한 기능을 가진 단백질이다. 자기 자신은 변하지 않으며 대부분의 생명 활동과 대사 활동에 관여하고 있는 생체 촉매이다. 이 특수한 단백질 덩어리가 매개하는 생화학 반응은 수천 가지며 효소의 기능을 이해하고 체계적으로 분류하는 것은 생명 현상 연구에 있어 아주 중요하다.

효소 번호는 1961년 국제 생화학 연합의 효소 위원회(Enzyme Commission, EC)에 의해 제안된 효소 분류 체계로 사람이 직접 수행한 실험을 통해 확인된 촉매 반응을 따라 일정한 규칙을 가지고 효소 번호를 부여한다. 일련번호는 마침표로 구분된 4자리의 숫자로 표기하는데 각 자리의 숫자는 특정 단백질의 기질 및 반응 형식을 의미하며 네 자리의 조합을 통해 효소가 가진 특수한 기능을 요약한다. Fig. 1은 효소 번호의 구성을 나타내고 있다. 주요 기능을 나타내는 것은 네 자리 중 앞에서부터 세 자리까지이다. 효소 번호가 나타내는 각 자리의 의미는 다음과 같다.

가장 첫 번째 자리는 대분류(Main class)로 효소가 담당하는 주요 기능 7가지를 나타낸다. 산화환원 기능, 전달 기능, 가수분해 기능, 분해(기제거) 기능, 이성질화 기능, 연결(합성) 기능, 막 수송 및 막 분리 기능이 이에 해당된다. 두 번째 자리인 중분류(Sub class)와 세 번째 자리인 세분류(Sub-sub class)는 대분류의 반응을 더 세분화하여 부여한다. 기능에 따라 나타내는 작용이 완전히 달라지므로 중분류, 세분류의 분류 지표는 대분류를 따라서 조금씩 다르고 약간의 의미 차이를 보인다. 네번째 자리는 고유 식별자 (Serial number)로 세분류 내에서의 일련번호를 나타낸다. 이 번호는 효소 위원회에서만 효소를 등록할 때 매길 수 있다.

효소를 해당하는 기능에 따라 규칙적 표기로 정리한 효소 번호 데이터는 체계적으로 정립된 역사가 길고 데이터의 신뢰도가 높아 딥러닝을 이용한 단백질 기능 예측 연구에 사용되기 적합하다.

2.2 CNN

합성곱 신경망(Convolutional Neural Network; CNN)은 이미지와 영상 처리에 특화된 인공신경망으로 합성곱 층

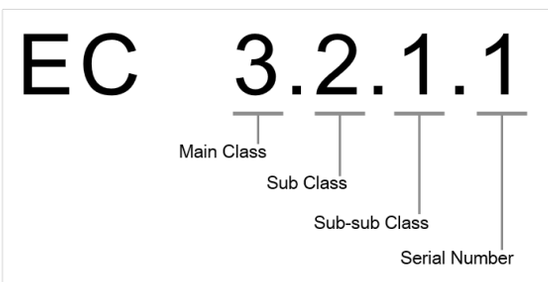

Fig. 1. EC Number Structure

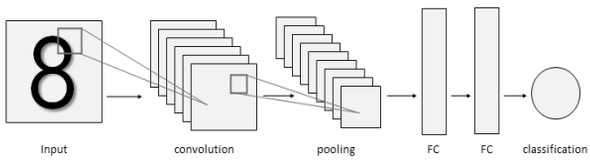

Fig. 2. CNN Structure

(Convolution Layer), 풀링 층(Pooling Layer), 완전 연결 계층 순으로 구성되며 합성곱 층과 풀링 층을 통해 특정 필터 내지는 커널(Kernel)이라고 불리는 정해진 합성곱 범위의 특징(Feature) 정보를 추출한다. 그 구조는 Fig. 2와 같다. 합성곱 연산을 수행하는 동안 약 5배 정도 부분화된 데이터가 파생되므로 풀링 층을 통해서 피처의 크기를 줄이면서 파편화된 이미지 내의 패턴 정보를 반복해서 특정한다. 이 부분적인 이미지 정보들은 차후 합쳐지면서 이미지 자체가 아닌, 이미지 내의 불변하는 패턴 정보만을 담은 고유한 특징 맵(Feature map)으로 변환된다.

이미지의 공간적, 지역적 정보를 유지하면서 사소한 개별 특성을 제거하고 핵심 특징만을 추출하는 것이 주요 특징이다. 일정 규칙을 가진 데이터 처리에 적합하며 이를 이용한 문장 분석, 주가 예측과 같은 시계열 데이터에 적용하려는 시도가 왕성하다[17,18].

2.3 RNN

순환 신경망(Recurrent Neural Network; RNN)은 시퀀스 길이와 관계없이 입력과 출력을 다룰 수 있는 체인형 네트워크 구조로 모델에 들어오는 값을 누락 없이 순서대로 처리한다. 시간적 순서가 중요한 음성이나 문자와 같은 시계열 데이터 처리에 특화된 인공신경망이다. 그 구조는 Fig. 3과 같다. 내부의 순환구조를 통해 자기 순환을 반복하며 과거의 학습 값을 현재의 학습 값에 지속적으로 반영한다. 내부의 순환구조를 담당하는 노드는 셀(Cell)이라고 부르며 은닉층에서 활성화 함수를 통해 입력 벡터와 출력 벡터 사이에서 하나의 가중치를 계속 갱신하고 이전의 값을 기억해두는 기억 저장소 역할을 함께 수행한다.

RNN에는 과거의 학습 결과가 사라지는 장기 의존성 문제(Long-term dependency problem)가 존재한다. RNN만으로 이 문제를 해결하기 어려운 이유를 S. Hochreiter(1991)와 Y. Bengio(1994)가 증명하였다[19,20].

1) LSTM

장단기 메모리(Long Short-Term Memory; LSTM)는 RNN 모델의 장기 의존성 문제를 해결하기 위해 제안된 모델 중 하나이다. 새로 들어온 입력 값뿐만이 아닌, 재귀 되어 돌아온 이전의 출력 값을 잊을지, 얼마나 받을지 게이트로 결정한다. 구조적으로 RNN을 닮아있으나 3개의 게이트로 셀의 상태를 관리함으로써 반복된 곱 연산으로 인한 셀 데이터의 기울기 소실(Vanishing gradient) 문제를 해결하고 과거의 학습 영향을 거리가 먼 단어까지 전달할 수 있게 되었다. LSTM의 셀이 과거의 정보를 얼마나 기억할 것인지, 현재의 정보를 얼마나 기억할 것인지는 sigmoid 함수로 결정된다. 값이 1이면 모든 정보를 보존하고, 0인 경우 모든 정보를 잊는다.

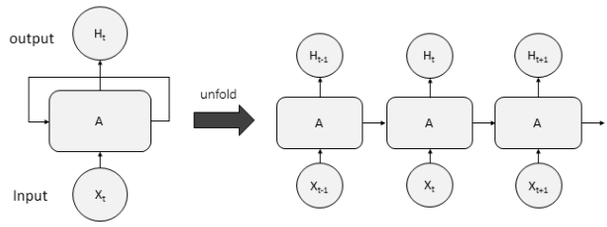

Fig. 3. RNN Structure

2) GRU

게이트 순환 유닛(Gated Recurrent Unit; GRU)은 2014년에 K. Cho et al[21] 에 의해 고안된 모델이다. LSTM 모델의 경량형에 가깝다. LSTM의 마지막 단계인 출력 게이트가 사라지고 망각 게이트와 입력 게이트가 하나로 합쳐진 갱신 게이트(Update Gate)와 망각 게이트와 유사한 역할을 하는 초기화 게이트(Reset Gate)가 모델의 셀을 구성한다. 초기화 게이트는 과거 학습 상태를 얼마나 반영할지, 갱신 게이트는 이전 상태와 현재 상태를 얼마나 반영할지를 결정한다. 순차연산이기에 병렬처리가 어렵고 연산량이 늘어날수록 학습 속도가 느려질 수 있는 LSTM에 비해 GRU는 중간에 처리할 파라미터 연산량이 줄어드는 장점이 있다.

3. 실험 방법

3.1 Dataset

모델이 예측하는 문제는 두 가지이다. 8,578가지의 시험 데이터 입력을 사용하여 효소와 비효소인지를 예측하는 이진 분류 문제와 약 60,445가지의 시험 데이터 입력을 사용하여 17개의 효소 번호를 예측하는 다중 분류 문제이며 이를 위해 두 가지 데이터 셋을 사용하였다.

1) 효소(Enzyme)와 비효소(Non-enzyme)

효소와 비효소 데이터를 사용하여 실험하였다. 데이터는 BRENDA(https://www.brenda-enzymes.org/)에서 오픈 데이터를 파싱하여 사용하였다. 데이터 파싱 때의 업데이트 날짜는 2018.10이었다. 비효소 데이터 수가 효소 데이터에 비해 현저히 적어 효소 데이터 셋에서 랜덤하게 뽑아서 비효소 데이터 셋과 비슷한 수로 맞춰줄 필요성이 있었다. 다운샘플링(Down-sampling)을 실시하기 전에 과적합을 방지하기 위해 효소와 비효소 두 데이터 모두 6:2:2 비율로 분할하

Table 1. Imbalanced Class Distribution after Random-sampling

EC	All	1st Train 0.056%	2nd Train 0.056%	Valid 0.032%	Test 0.032%
3	53	2	1	-	-
3.1	2,314,776	77,461	77,695	14,779	14,812
3.2	702,941	23,352	23,443	4,537	4,527
3.3	39,329	1,313	1,292	245	261
3.4	895,324	30,243	30,080	5,763	5,725
3.5	1,124,187	37,793	37,976	7,192	7,280
3.6	826,589	27,780	27,790	5,387	5,263
3.7	33,282	1,170	1,103	214	226
3.8	7,965	214	269	53	53
3.9	374	16	12	1	3
3.10	53	-	-	-	-
3.11	5,962	216	196	38	36
3.12	5	-	-	-	-
3.13	638	24	20	1	2
7	2	-	-	-	-
7.1	3,140,836	105,976	105,701	20,050	20,019
7.2	134,104	4,400	4,481	818	832
7.3	74,573	2,507	2,521	480	471
7.4	24,618	833	823	144	164
7.5	39,121	1,306	1,366	251	249
7.6	82,949	2,871	2,708	512	522
Total	9,447,681	317,477	317,477	60,465	60,445

여 각각 훈련(train), 검정(valid), 시험(test) 셋을 구성하였다. 이후 효소 훈련 세트에서 랜덤하게 뽑아 비효소 훈련 세트의 데이터 개수와 동일하게 맞추어 병합하였고, 검정, 시험 세트도 각각 같은 과정을 거쳤다. 실험에 사용된 두 데이터의 수는 5대 5로 완전히 동일하며 최종 데이터 셋은 훈련 데이터 24,839개, 시험 데이터 8,578개, 검정 데이터 8,578개이다. 모든 시퀀스의 길이는 50 이상 1,000 이하이다. 이 데이터 셋은 1번이라고 지칭하였다.

2) 효소 번호(EC number) 3번과 7번

BRENDA에서 제공한 오픈 데이터를 파싱하여 사용하였고 그 중 효소 번호 대분류(main class)가 3번과 7번인 데이터를 이용하여 실험하였다. 효소 번호 7번은 3번에서 분리되어 나온 지 얼마 되지 않아 많은 모델에서 아직까지도 3번으로 분류되곤 한다. 선행 연구인 EnzyNet(2018), MF-EFP(2020), DeEPn (2020) 에서는 효소 번호를 대분류까지 예측하고, UDSMProt (2020)에서는 중분류(sub class)까지 예측하는 점을 고려하여[2-5] 본 연구에서는 대분류부터 중분류까지만 고려되었다.

클래스 라벨(Class label)은 각각 '3', '3.1', '3.2', '3.3', '3.4', '3.5', '3.6', '3.7', '3.8', '3.9', '3.10', '3.11', '3.12', '3.13', '7', '7.1', '7.2', '7.3', '7.4', '7.5', '7.6'으로 총 21개였으나 라벨 '3', '3.10', '3.12', '7'은 BRENDA에서 받아온 샘플데이터 자체가 몹시 희귀한 편에 속하므로 학습 중에 아예 무시되었다. 따라서 실제로 분류되는 클래스는 총 17개이다.

제시된 Table 1의 All 열은 불균형 데이터 셋의 실제 샘플 클래스 분포도를 나타낸 것이다. 1번 데이터 세트인 효소와 비효소 데이터 셋과는 달리 다운 샘플링 되지 않았고 각 클래스 간의 수 차이가 명확하고 불균형하다.

과적합을 방지하기 위해 총 9,447,681개의 시퀀스 데이터를 각각 6:2:2로 나누어 Train_60, Valid_20, Test_20 데이터 셋으로 분화시켰다. 이후 각 데이터 셋마다 따로 랜덤 샘플링을 실시하였다. 최종 훈련데이터는 Train_60으로부터 0.056% 랜덤하게 선택되고 최종 시험 데이터와 최종 검증 데이터는 각각 Valid_20, Test_20에서 0.032% 씩 동일한 비율로 랜덤하게 선택되었다. 이것을 3회씩 반복했을 때 라벨 '3', '3.10', '3.12', '7'을 제외한 클래스는 본래 데이터 개수에 비례하여 항상 비슷한 비율로 샘플링 되는 것을 확인하였다. 이를 Table 1의 1st Train 0.056% 2nd Train 0.056%, Valid 0.032%, Test 0.032% 4개의 열과 Fig. 4의 막대 그래프로 나타내었다.

이를 통해 모델 학습의 균일성을 확보하였으며 반복 실험하더라도 샘플 개수가 달라지는 일이 없기에 늘 균일한 실험 결과를 얻을 수 있음을 확인하였다.

각 데이터 셋 내에 존재하는 모든 시퀀스의 길이는 첫 번째 데이터 셋 기준과 동일하게 50 이상 1,000 이하다. 이 데이터 셋은 2번이라고 지칭하였다.

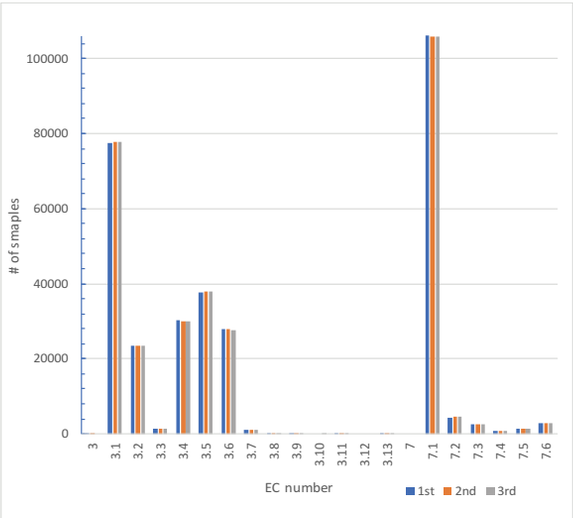

Fig. 4. Comparison Graph of 3 Train Datasets Class Distribution after Random-sampling

3.2 Embedding Matrix

단백질 서열(Protein Sequence)은 약 20개의 아미노산 알파벳 표현과 5개의 특별한 문자 표현으로 이루어져있다. 이 문자열을 모델에 넣어주기 위해서는 원-핫 인코딩(One-Hot Encoding)과 같은 벡터화가 반드시 필요하다. 원 핫 인코딩은 표현하고 싶은 단어의 인덱스에 1을 부여하고, 나머지 인덱스에는 0을 부여하는 표현 방식이다. 이렇게 표현된 벡터를 원-핫 벡터(One-Hot vector)라고 한다.

5개의 특별한 문자표현은 단백질 서열을 이용한 예측 연구 분야에서 '-'또는 'x'로 치환되어 약 21개의 문자로 자주 표현된다. DeepEC[6], DeepACLSTM[9], Min et al[12] 등에서도 유사하게 다루었으며 논문에서도 똑같이 X로 치환하는 방식으로 제외하였다. 원 핫 인코딩을 거친 뒤 1000x21의 규격으로 임베딩 레이어에 넣어주었으며 21은 아미노산 알파벳 표현, 1000은 시퀀스 길이 제한을 나타낸다. 이렇게 제작된 원 핫 인코딩 임베딩 매트릭스(Embedding Matrix)는 모든 모델에 동일하게 적용되었다.

3.3 심층 신경망의 학습 구조

이 논문에서 등장하는 모델들은 같은 시퀀스 데이터를 가지고 같은 방법으로 임베딩 된 매트릭스를 가져가 각각의 알고리즘으로 학습한다. 이후 같은 조건의 완전 연결 계층을 통과하며 저마다 연산된 뒤에 최종적으로 분류 결과를 내놓는다. 첫 번째 데이터 셋은 2개의 분류 결과를 내고, 두 번째 데이터 셋은 총 17개의 분류 결과를 내게 된다. 그 구조와 흐름은 Fig. 5와 같다.

1) CNN

CNN 모델은 필터 개수가 동일한 단일 모델과 중첩 모델로 두 가지로 구성된다. 두 모델 전부 1D CNN을 사용하며 필터 개수는 128개이다. 단일 모델은 Convolution1D 와 1D MaxPooling으로 이루어진 컨볼루션 층과 512개 노드를 가지는 두 개의 히든 레이어로 분류 모델을 구성하였다.

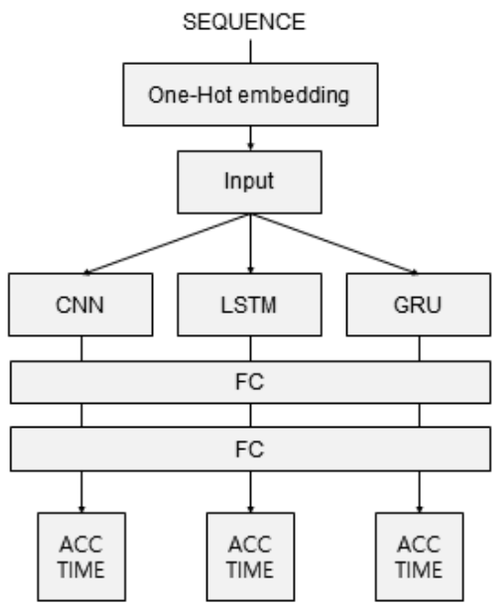

Fig. 5. Flow-diagram of Test Models

중첩 모델은 단일 모델과 구성은 같되 Y. Kim(2014)에서 수행한 문장 분류를 위한 컨볼루션 신경망 구조[22]를 취하고 있으며 1D CNN의 필터 3가지를 중첩하는 컨볼루션 층과 1D MaxPooling, 512개 노드를 가지는 두 개의 히든 레이어로 구성하였다. 필터 중첩 모델은 단백질 기능 예측 페이퍼인 DeepEC에서도 같은 방식으로 사용된 바 있다.

필터 크기(Filter size)는 연구마다 제각각이나 CNN에 사용되는 필터 크기는 2~9 범위를 잘 넘지 않았고 드물게 중첩 필터를 사용하는 DeepEC에서 16이 적용된 것이 확인되었다. 따라서 단일 모델에서는 2-9까지의 필터 크기를 적용하여 실험하고 가장 결과가 좋았던 것을 선별하였다. 중첩 모델에서는 Y. Kim(2014)에서 사용된 (3,4,5) 중첩 필터와 DeepEC에서 (4,8,16) 중첩 필터, 마지막으로 단일 모델에서 가장 결과가 좋았던 크기 3개를 조합하여 마지막 중첩 필터를 정하였다.

2) LSTM and GRU

단백질 예측 분야에서 LSTM과 GRU는 unit 150~ 300개 사이로 2회에서 최대 6회까지 심층 심경망을 쌓는 경향을 보였다. 컴퓨터 성능을 고려하여 LSTM과 GRU의 심층 신경망의 구성을 정하기 위해 적절한 유닛 수와 깊이를 선별하기 위한 실험을 수행하였다. 1번 데이터 셋을 대상으로 심층 신경망 깊이에 따른 단일 모델의 성능을 확인한 후 2번 데이터 셋

에는 가장 결과가 좋았던 모델 값만 적용하였다. 이후 CNN과 동일하게 512개 노드를 가지는 두 개의 은닉층으로 분류 모델을 구성하였다.

3.4 Learning Rate(학습률)과 Batchsize(배치크기)

전체 표본 수가 훨씬 적은 1번 데이터 셋의 학습률은 0.01부터 0.0001까지만 고려되었고, 실험을 통해 선별하였다. 전체 표본 수가 훨씬 많은 2번 세트의 학습률은 클래스가 불균형하고 각 클래스별 표본 수는 오히려 1번 세트보다 적은 경우가 많아 모델이 복잡해질수록 과적합이 일어나기 쉬웠다. 따라서 학습률을 아주 작게 적용했으나 RNN계열 모델에서 1번 데이터 셋에 적용했던 학습률로는 충분하지 않아 0.000001까지 설정하여 실험을 재수행 하였다.

배치 크기는 32로 모두 동일하게 설정하였다. D. Masters and C. Luschi가 32 크기 이상의 배치는 오히려 학습을 저해한다는 보고를 한 적이 있어[23] 1번 데이터 셋에 대해 16, 32, 64로 각각 실험해본 뒤 모델 학습 속도를 고려하여 32로 최종 결정하였다. 조기종료(Early Stopping)을 적용하긴 하였으나 기본 epoch는 모두 30으로 설정되어있다.

3.5 학습 진행

모든 실험은 동일하게 윈도우 10에서 Python 3.8.8, RAM 8GB, GPU NVIDIA GeForce GTX 1660 SUPER, Tensorflow 2.5 + Keras-lr + Jupter notebook(Chrome) 환경에서 동작하였다.

과적합을 막기 위해서 모든 모델에 배치 정규화(Batch Normalization)와 조기종료(Early Stopping)를 적용하였다. 조기 종료는 검증 손실(Validation loss)을 기준으로 판단하며 바로 직전 값과 비교하여 총합 5회 성능의 향상이 보이지 않을 시 5번째에 바로 종료하도록 설정하였다.

4. 실험 결과 및 평가

4.1 RNN 계열 유닛(Unit) 선정 실험

Table 2는 실험 결과이다. 두 RNN 계열의 모델에 적용될 학습률은 0.0001로 선정하였다. 앞서 CNN에서 고려되었던 학습률 0.01, 0.001은 RNN 계열의 모델에 적용하자 1 epoch 학습 중에 너무 빠른 과적합이 일어나 LSTM과 GRU 유닛 선정 실험에서 제외되었다. LSTM과 GRU 모두 사용되는 유닛 수가 많을수록 정확도가 점점 증가하는 추세를 보였다. LSTM은 깊이가 3을 초과하는 순간부터 정확도가 떨어지기 시작했으나 GRU는 2를 초과 하는 순간부터 정확도가 떨어지기 시작했다. 가장 정확도가 높았던 것은 83.4%로 유닛 50을 3회 쌓은 LSTM이었고 GRU에서는 유닛 50을 2회 쌓은 것이 약 82.5%의 정확도로 다른 조건의 GRU 보다 그 결과가 좋았다.

그러나 유닛 수가 50을 넘어가면서부터 모델의 깊이가 깊어지자 과도한 메모리 적재로 빈번히 커널이 강제 종료되는 문제가 발생하였다. 메모리 부담과 정확도를 고려하면 50 유닛을 2회 또는 3회 쌓는 것이 두 모델에 동일하게 적용하기에 가장 좋았고 보다 원활한 실험수행을 위하여 커널 종료 위험성이 큰 3회 쌓기 대신 LSTM과 GRU에서 약 82%의 균일한 결과를 내는 50 유닛 2회 쌓기로 결정하였다.

Table 2. The First Dataset Accuracy of The LSTM and GRU Model according to The Depth with Each Unit

Learning Rate		0.0001	
units	depth	LSTM	GRU
20	1	0.8210	0.7908
	2	0.8140	0.7878
	3	0.8172	0.7700
	4	0.6225	0.6797
	5	0.8107	N/A
	1	0.8214	0.8080
50	2	0.8260	**0.8256**
	3	**0.8341**	**0.8233**
80	1	0.8205	0.8112
	2	0.8309	0.8162
100	1	0.8309	0.8083

4.2 1D CNN 필터 크기 선정 실험

Table 3는 1번 데이터 셋의 필터 크기에 따른 각 모델의 정확도 평가를 나타내고 있다. Table 3번에서 학습률(Learning rate)은 Lr로, 정확도(Accuracy)는 ACC로, 손실(Loss)는 Loss로 표기하였다.

학습률이 작아질수록 정확도는 그 전 학습률에 비해서 성능이 떨어지는 모습을 보였으며, 0.01일 때 가장 좋았다. 또한 필

Table 3. Accuracy of Single 1D CNN according to Filter Size

Lr	0.01		0.001		0.0001	
Filter size	ACC	Loss	ACC	Loss	ACC	Loss
2	0.7971	0.4685	0.7842	0.5835	0.7747	0.5269
3	0.8279	0.4357	0.8016	0.5843	0.7917	0.5078
4	0.8201	0.4269	0.8157	0.5461	0.7743	0.5750
5	0.8244	0.4736	0.8118	0.5944	0.7757	0.5735
6	0.8208	0.4989	0.8039	0.5967	0.7789	0.6284
7	0.8129	0.5213	0.8171	0.5756	0.7731	0.5933
8	0.8157	0.5906	0.8161	0.6279	0.7750	0.6488
9	0.8250	0.5469	0.8222	0.5770	0.7807	0.6365

Table 4. Accuracy of Complex 1D CNN according to Filter Size

Lr	0.01		0.001		0.0001	
Filter size	ACC	Loss	ACC	Loss	ACC	Loss
3, 4, 5	0.8268	0.5528	0.8307	0.5973	0.8016	0.6155
4, 8, 16	0.8030	0.6231	0.8327	0.6318	0.7971	0.8453
3, 5, 9	0.8335	0.6340	0.8254	0.5892	0.8027	0.7469

터 크기가 커질수록 손실 값이 같이 증가하는 경향을 보여주었다. 그중 3, 4, 5, 7, 9 크기의 결과가 좋았고, 3가지 학습률에 평균적으로 좋은 정확도를 낸 크기는 9였다. 짝수 크기일 때보다 홀수 크기일 때의 결과가 좋았고, 필터 5, 7 크기 중에는 5 크기 필터의 결과가 7 크기와 비슷한 정확도를 낸 것에 비하여 손실 값은 약 0.1-0.2% 더 적었다. 이 결과를 토대로 중첩 모델에는 (3,5,9) 조합의 필터가 추가로 실험되었다.

Table 3에 포함하지는 않았으나 학습률이 0.00001, 0.000001 정도로 아주 작은 경우 훈련과 검증의 손실 값이 비슷하게 줄어들어 아주 안정적인 학습이 가능했다. 그러나 오히려 정확도는 크게 향상되지 않았고 어느 크기의 필터를 사용하던 77%로 일정했으며 각 필터 크기 간의 정확도와 손실 오차는 0.01% 내외로 다른 실험에 비해 미진한 학습 결과를 산출하는 것을 확인하였다.

Table 4는 중첩 모델에 사용된 필터 크기와 학습률에 따른 정확도와 손실 값을 나타낸 것이다. 짝수로만 이루어진 (4,8,16) 중첩 모델의 결과가 0.01에서 80%, 0.001에서 83%, 0.0001에서 79%로 가장 좋지 않았고, 홀수로만 이루어진 (3,5,9)의 정확도는 높았으나 0.001 Lr 에서의 결과를 제외하고는 비슷한 정확도를 낸 (3,4,5) 중첩 모델보다 손실 값이 항상 더 컸다.

4.3 단일 모델 실험 결과

1번 데이터 셋을 대상으로 수행한 유닛 선정 실험과 필터 크기 선정 실험 결과는 2번 데이터 셋에 적용한 임의 실험 결과와 비슷하였기에 단일 모델 실험과 결합 모델 실험에도 그대로 적용되었다.

LSTM과 GRU는 50 유닛씩 2회 중첩되었고 과적합 문제로 인해 CNN과는 다른 학습률이 적용되었다. 단일 CNN에는 앞선 실험 결과에서 좋은 성과를 보여주었던 학습률 0.01과 필터 크기 9가 사용되었고 필터 중첩 모델인 중첩 CNN에는 (3,4,5)가 사용되었다.

평가척도는 클래스 간의 불균형을 고려하여 마이크로 정밀도, 재현율, F1 스코어로 정하였고 정확도(ACC)와 손실(Loss) 값이 포함되었다. Table 5의 Test Time은 시험 데이터(0.032%) 전체를 예측하는데 소요된 총 시간을 나타내고, Training Time은 배치 크기 32로 1 epoch당 걸린 학습 속도를 나타내고 있다.

Table 5. Experimental Result about Dataset 1

Model	Single CNN	Complex CNN	LSTM	GRU
Lr	0.01		0.00001	
ACC	0.8263	0.8202	0.8258	0.8049
Loss	0.5607	0.5867	0.4798	0.5631
Training Time	16s	21s	50s	47s
Test Time	11s	14s	139s	134s
Recall	0.826	0.820	0.826	0.806
Precision	0.826	0.820	0.826	0.806
F1_score	0.826	0.820	0.826	0.806

1) 1번 데이터 셋 - 효소와 비효소

Table 5는 1번 데이터 셋에 적용된 각 모델의 실험 결과 값을 나타내고 있다. 효소와 효소가 아닌 것을 구분하는 문제에서 가장 높은 성능을 보인 것은 LSTM 모델이었다.

LSTM의 정확도는 82.4%로 중첩 CNN에서 보인 82.3%의 정확도와 유사하였으나 그보다 훨씬 손실 값이 적었다. CNN 단일 모델과 중첩 모델에 비교하면 RNN의 계열 모델들인 LSTM과 GRU의 학습 속도는 약 2배 느렸고, 예측 속도는 약 10배 더 소요되었다. 가장 결과가 좋지 않았던 것은 GRU로 LSTM과 학습과 예측에 걸리는 속도는 비슷하나 정확도와 손실 값은 CNN으로 구성된 두 모델과 유사했다.

2) 2번 데이터 셋 - 효소 번호 3번과 7번

Table 6은 2번 데이터 셋에서의 각 모델의 실험 결괏값을 나타내고 있다. 1번 데이터 셋에서 사용했던 학습률로는 과적합이 너무 이르게 일어났기에 모든 모델에서 기존에 사용되었던 학습률보다 더 작은 값을 적용하여 실험하였다. RNN 계열 모델

Table 6. Experimental Result about Dataset 2

Model	Single CNN	Complex CNN	LSTM	GRU
Lr	0.00001		0.000001	
ACC	0.8928	0.9079	0.9166	0.8942
Loss	0.3386	0.2973	0.2888	0.3825
Training Time	172s	250s	604s	561s
Test Time	71s	95s	982s	962s
Recall	0.892	0.907	0.916	0.894
Precision	0.892	0.907	0.916	0.894
F1_score	0.892	0.907	0.916	0.894

Table 7. Ensemble Models Results

	Dataset 1		Dataset 2	
Model	CNN-LSTM	CNN-GRU	CNN-LSTM	CNN-GRU
Lr	0.00001		0.000001	
ACC	0.8130	0.7927	0.9146	0.9000
Loss	0.6263	0.7249	0.2919	0.3402
Training Time	52s	47s	607s	590s
Test Time	139s	134s	993s	970s
Recall	0.81	0.79	0.91	0.90
Precision	0.81	0.79	0.91	0.90
F1_score	0.81	0.79	0.91	0.90

인 LSTM과 GRU 모두 0.00001 학습률로 실험시 epoch 10에 도달하자 훈련 손실 값이 0에 수렴하여 사라지므로 과적합을 피하기 위해 더 작은 값이 적용되어야 할 필요성이 있었다.

LSTM의 성능이 약 91%로 가장 높은 성능을 보였고 손실 값도 가장 낮았으나 학습 시간과 예측 테스트 시간이 604s, 982s로 가장 길었다. 그 다음으로 좋은 결과를 낸 것은 정확도 약 90%의 중첩 CNN이었다. 손실이 적고 학습 시간과 예측 테스트 시간 모두 RNN 계열 모델보다 빨랐다. RNN 계열 모델에서 가장 학습 속도가 빠른 것은 GRU였고 두 모델의 예측 시간은 20s 정도로 차이가 났다. LSTM과 GRU 모두 예측 속도가 CNN으로 구성된 두 모델에 비하여 약 10배 정도 증가했다.

가장 좋은 결과는 LSTM이나 모델의 학습 속도와 예측 속도를 고려한다면 중첩 모델인 Complex CNN을 사용하는 것이 더 유용할 수 있음을 확인하였다.

4.4 결합 모델 실험 결과

Table 7은 결합 모델을 실험한 결과이다. 선행 연구로 중첩 CNN 모델과 RNN 계열의 결합은 확인되지 않았기에 실험에서 제외되었고 오로지 단일 CNN과 LSTM, GRU를 조합하여 진행하였다. 마찬가지로 단일 CNN의 필터 크기는 9로 고정되었다.

CNN-LSTM 모델이 두 데이터 분류 문제에서 가장 우수한 성능을 보였고 단순 분류 문제인 1번 데이터 셋보다 복잡한 분류 문제인 2번 데이터 셋에서 더 정확도가 높았다. CNN-LSTM 모델이 두 문제에서 모두 정확도가 1.5% 더 높았고 손실값은 0.5% 정도 더 낮았다. 학습 속도와 예측 속도는 두 결합 모델이 비슷하였으나 CNN-GRU 모델이 CNN-LSTM 모델보다 학습 속도에서 약 5초, 약 15초 정도 더 빠르고 예측 속도에서 약 5초, 약 23초 정도로 더 빨랐다. 다만 성능 평가에 고려될 만큼 유의미한 수치는 아니었다.

4.5 결과 분석

Table 2의 유닛 선정 실험을 통하여 유닛의 개수와는 상관없이 심층 신경망의 깊이가 깊어질수록 오히려 모델의 성능이 떨어지는 것을 확인하였고, Table 3의 필터 선정 실험으로 CNN 이용 시 데이터의 특징을 추출할 때 사용할 필터의 크기는 짝수보다는 홀수인 것이 적합한 것과 큰 필터보다는 작은 필터를 사용하는 것이 시퀀스 데이터 처리에 더 적합한 것을 확인하였다.

1번 데이터 셋을 통하여 효소와 효소가 아닌 단백질 데이터 간에는 서열의 패턴적 차이가 있음을 확인할 수 있었고, 2번 데이터 셋과 같이 둘 다 효소이지만 기능적 차이가 있는 경우에도 서열상으로 패턴적 차이를 보임을 선행 연구뿐만 아니라 본 실험으로도 재확인할 수 있었다.

Table 5, 6, 7을 통해 단순 분류 문제에서 단일 모델로 LSTM의 성능이 준수하고, 복잡한 분류 문제에서는 단일 모델로 중첩 CNN이 더 적합하며, 결합 모델로 CNN-LSTM의 연계 모델이 상대적으로 더 우수함을 확인하였다.

그러나 학습 속도, 예측 속도를 제외하고 순수 정확도만을 따진다면 모든 모델에서 LSTM의 결과가 더 우수하였는데 이는 LSTM이 순서가 있는 정보, 단어 간에 연관성이 존재하는 데이터를 다룰 때 이점이 있는 모델이기 때문으로 보인다. 단백질 서열은 순서를 가진 정보로써 DNA가 체내에서 단백질로 해독될 때 정해진 순서에 의해서 조합되는데 이렇게 구성된 아미노산 서열에는 단백질 기능에 대한 정보가 포함된다. 본 실험 결과에서는 LSTM 모델이 아미노산 간의 관계나 연결성으로 인해 결정되는 특정 기능에 대한 패턴을 다른 모델에 비해 더 잘 파악하는 것으로 확인되었다.

또한, 같은 RNN 기반의 모델이라고 하더라도 서열정보를 파악하는 데 있어 GRU와 LSTM의 사이에는 분명한 성능적 차이가 존재했으며, 같은 조건 아래 GRU보다는 LSTM을 사용하는 것이 단백질 서열 내에 존재하는 기능과 관련된 미세한 패턴을 찾아내는 작업에 더 적합함을 확인하였다.

5. 결 론

본 논문에서는 단백질의 서열을 이용하는 딥러닝 분야에서 가장 많이 사용되는 대표 딥러닝 모델인 CNN, LSTM, GRU 모델과 이를 이용한 두가지 결합 모델에 동일 데이터를 적용하여 각 알고리즘의 단일 모델 성능과 결합 모델의 성능을 정확도와 속도를 기준으로 비교 평가하였으며 최종 평가 척도를 마이크로 정밀도, 재현율, F1 스코어로 나타내었다.

본 연구를 통해 서열 데이터를 다룰 때는 LSTM이 가장 우수함을 확인하였고, CNN과 RNN 계열의 결합을 진행하면 단백질 서열의 핵심 정보는 그대로 유지하되 피처의 크기를 줄여 첫 번째 완전 연결 계층에서 폭발하는 연산량을 줄일 수 있어 유용함을 확인하였다.

알고리즘을 직접 비교하기 위하여 본 논문에서 단순화하여 비교실험을 수행하였으나 본 논문에서 보인 단일 CNN의 예외적인 결과처럼 앞선 실험에서 모델의 성능이 좋지 않았음에도 다른 데이터 셋에서 다른 학습률을 적용할 때 크게 좋은 결과를 보이는 경우도 다수이다. 따라서 시퀀스 임베딩을 대상으로 한 모델의 학습 구조와 학습률, 필터 크기와 유닛에 따른 학습의 변화는 앞으로 더 많이 연구되어야한다.

본 연구결과를 기반으로 향후 연구에서는 다른 선행 연구에서 적용하지 않았던 중첩 CNN-LSTM의 연계를 이용한 효소 기능 예측 모델 연구를 수행하고자 한다.

References

[1] A. Dalkiran, A. S. Rifaioglu, M. J. Martin, R. Cetin-Atalay, V. Atalay, T. Doğan, "ECPred: A tool for the prediction of the enzymatic functions of protein sequences based on the EC nomenclature," *BMC Bioinformatics*, Vol.19, No.1, pp.334, 2018.

[2] A. Amidi, S. Amidi, D. Vlachakis, V. Megalooikonomou, N. Paragios, E. I. Zacharaki, "EnzyNet: Enzyme classification using 3D convolutional neural networks on spatial representation," *PeerJ*, Vol.6, pp.e4750, 2018.

[3] X. Xiao, L. Duan, G. Xue, G. Chen, P. Wang, W. R. Qiu, "MF-EFP: Predicting multi-functional enzymes function using improved hybrid multi-label classifier," *IEEE Access*, Vol.8, pp.50276-50284, 2020.

[4] N. Strodthoff, P. Wagner, M. Wenzel, and W. Samek, "UDSMProt: Universal deep sequence models for protein classification," *Bioinformatics*, Vol.36, Iss.8, pp.2401-2409, 2020.

[5] R. Semwal, I. Aier, P. Tyagi, and P. K. Varadwaj, "DeEPn: A deep neural network based tool for enzyme functional annotation," *Journal of Biomolecular Structure and Dynamics*, Vol.39, No.8, pp.2733-2743, 2021.

[6] J. Y. Ryu, H. U. Kim, and S. Y. Lee, "Deep learning enables high-quality and high-throughput prediction of enzyme commission numbers," *Proceedings of the National Academy of Sciences*, Vol.116, No.28, pp.13996-14001, 2019.

[7] S. A. Memon, K. A. Khan, and H. Naveed, "HECNet: A hierarchical approach to enzyme function classification using a Siamese Triplet Network," *Bioinformatics*, Vol.36, No.17, pp.4583-4589, 2020.

[8] C. Mirabello and B. Wallner, "rawMSA: End-to-end deep learning using raw multiple sequence alignments." *PloS one*, Vol.14, No.8, pp.e0220182, 2019.

[9] Y. Guo, W. Li, B. Wang, H. Liu, and D. Zhou, "DeepACLSTM: Deep asymmetric convolutional long short-term memory neural models for protein secondary structure prediction," *BMC Bioinformatics*, Vol.20, No.1, pp.341, 2019.

[10] E. C. Alley, G Khimulya, S. Biswas, M. AlQuraishi, G. M. Church, "Unified rational protein engineering with sequence-based deep representation learning," *Nature Methods*, Vol.16, No.12, pp.1315-1322, 2019.

[11] Y. Jiang, D. Wang, and D. Xu, "DeepDom: Predicting protein domain boundary from sequence alone using stacked bidirectional LSTM," *Pacific Symposium on Biocomputing: Pacific Symposium on Biocomputing*, Vol.24, pp.66-75, 2019.

[12] S. Min, H. Kim, B. Lee, and S. Yoon, "Protein transfer learning improves identification of heat shock protein families," *PloS one*, Vol.16, No.5, pp.e0251865, 2021.

[13] C. Claudel-Renard, C. Chevalet, T. Farau, and D. Kahn, "Enzyme-specific profiles for genome annotation: PRIAM," *Nucleic Acids Research*, Vol.31, No.22, pp.6633-6639, 2003.

[14] R. d. O. Almeida and G. T. Valente, "Predicting metabolic pathways of plant enzymes without using sequence similarity: Models from machine learning," *The Plant Genome*, Vol.13, No.3, pp.e20043, 2020.

[15] N. Q. K. Le, E. K. Y. Yapp, and H. Yeh, "ET-GRU: Using multi-layer gated recurrent units to identify electron transport proteins," *BMC Bioinformatics*, Vol.20, No.1, pp.377, 2019.

[16] Z. Tao, B. Dong, Z. Teng, and Y. Zhao, "The classification of enzymes by deep learning," *IEEE Access*, Vol.8, pp.89802-89811, 2020.

[17] O. B. Sezer and A. M. Ozbayoglu, "Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach," *Applied Soft Computing*, Vol.70, pp.525-538, 2018.

[18] K. Bhardwaj, "Convolutional Neural Network(CNN/ConvNet) in stock price movement prediction," *arXiv:2106.01920*, 2021.

[19] S. Hochreiter, "Untersuchungen zu dynamischen neuronalen Netzen," Diplom thesis, Institut f Informatik, Technische Univ, Munich. 1991.

[20] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is diffcult," *IEEE Transactions on Neural Networks*, Vol.5, No.2, pp.157-166, 1994.

[21] K. Cho et al., "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[22] Y. Kim. "Convolutional Neural Networks for Sentence Classification". In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp.1746-1751, 2014.

[23] D. Masters and C. Luschi, "Revisiting small batch training for deep neural networks," Graphcore Research, *arXiv: 1804.07612*, 2018.

이 정 민

https://orcid.org/0000-0003-2237-4743
e-mail : starleejeung@gmail.com
2017년 선문대학교 컴퓨터공학부(학사)
2020년 ~ 현 재 선문대학교 컴퓨터융합전자공학과 바이오빅데이터융합 석사과정
관심분야 : Deep Learning in Bioinformatics

이 현

https://orcid.org/0000-0003-0089-1002
e-mail : mahyun91@sunmoon.ac.kr
2002년 선문대학교 전자계산학과(석사)
2010년 Univ. of Texas at Arlington Computer Science and Engineering 컴퓨터공학전공(박사)
2012년 ~ 현 재 선문대학교 컴퓨터공학부 부교수
관심분야 : 실시간 의사결정 시스템, 자율컴퓨팅, 휴먼케어 시스템, 사물인터넷 기반 가상물리(CPS)시스템