

지진 예측을 위한 머신러닝 기법 설계



김장환
전인혁
나건혁

지도교수 송길태

목 차

1. 요구조건 및 제약사항 분석 및 수정사항	3
1.1. 요구조건	3
1.2. 제약사항 분석	3
1.2.1. 제약사항	3
2. 설계 계획 변경 사항	4
2.1. CNN을 활용한 간단한 지진 탐지 모델 개발 (기존 계획)	4
2.2. CNN + LSTM으로 만든 지진 탐지 모델 개발	4
3. CNN 모델 적용 (기존 계획)	
3.1. Model 1 (Decision Tree)	5
3.1.1. 사용 데이터	5
3.1.2. DecisionTree Model 학습	7
3.2. Model 2 (Customized CNN Model)	9
3.3. Model 3 (-)	11
4. 모델 설명	5
4.1. 모델 소개	5
4.2. 목적	6
4.3. 모델 구조	6
4.4. 학습 방식	7
5. 결과 + 시각화	9
5.1. 샘플 데이터 다운로드	9
5.2. 샘플 데이터	14
5.3. 코드 및 시각화(Visualization)	14
6. 결론	25
7. 수행 체계	26
7.1. 구성원별 역할	26
7.2. 개발 일정	26
8. 참고 문헌	28

1. 요구조건 및 제약사항 분석 및 수정사항

1.1 요구조건

○ 지진 관련 데이터 분석

지진관련 데이터를 파악해 지진이 언제 어디서 발생하는지를 분석한다.

○ P파, S파 신호를 정확하게 검출

지진 신호를 비롯한 다양한 signal이 섞인 데이터에서 정확한 P파와 S파를 picking 해낸다.

○ 시그널 감지 활용

지반공사, 폭발 등의 신호와 지진 신호를 정확히 구분하여 조기 재난 경보에 활용할 수 있다.

○ 학습 요구사항과 소요 시간이 작은 모델 구현

1.2 제약사항 분석

1.2.1 제약사항

○ 학습 환경의 제한

=> 방대한 데이터량을 처리해야 하고, 머신러닝 특성상 오랜 시간 Training을 진행하기 위해 고성능의 GPU를 요구하지만 학부생의 장비 이용 및 서버 사용 비용이 제한적이고 개인 컴퓨터로는 한계가 있음.

=> google 의 Colab 이용, 하지만 딥러닝 모델을 훈련하기엔 사용 시간 제한과 한정적인 GPU와 메모리 사용.

○ 코로나-19로 인한 온라인 미팅 지향

=> 온라인 미팅 시 집중도 하락과 의사소통의 한계점이 있지만, 현 상황을 고려해 온라인 미팅을 지향하고, 필요시 방역 수칙을 준수하여 오프라인 미팅을 진행한다

2. 설계 계획 변경 사항

2.1 CNN을 활용한 가벼운 지진 탐지 모델 개발 (기존 계획)

- 세계적인 수준의 지진 탐지 모델들을 익히고, 간단한 CNN 기법을 활용한 자체적인 지진 탐지 모델을 설계한다.
- 기존 모델들에 비해 성능은 떨어지더라도 더 빠르게 결과값을 얻을 수 있는 모델을 설계한다.

2.2 CNN + Bidirectional LSTM으로 만든 지진 탐지 모델 개발

- 세계적인 수준의 지진 탐지 모델들을 익히고, 간단한 CNN 기법을 활용한 자체적인 지진 탐지 모델을 설계한다.
- CNN 기법은 이미지를 기반으로 2D Convolution을 진행하는 것이 일반적임.
- 하지만, 우리가 사용할 데이터셋은 해당 위치에서 발생한 지진 신호에서의 E(East), N(North), Z 3개 방향에 대한 데이터를 가지며, 각각의 신호 데이터들은 독립성을 가지므로, 각 신호에 독립적인 1D Convolution을 진행해야 하므로 일반적인 CNN 모델로는 어려움이 있음.
- 최고 수준의 모델의 성능을 따라갈 순 없겠지만, 간단하고 더 빠르게 작동하는 지진 탐지 모델을 개발

3. 개발 모델

3-1. Model 1 (Decision Tree)

- 사용 배경: Decision Tree: 각 분기 별 학습한 signal Data를 기준으로 Test set Signal을 판단해 지진과 Noise를 구분하는 Model 제작을 위해 사용한 알고리즘이다.

3-1-1. 사용 데이터

1) 데이터 가져오기 및 전처리

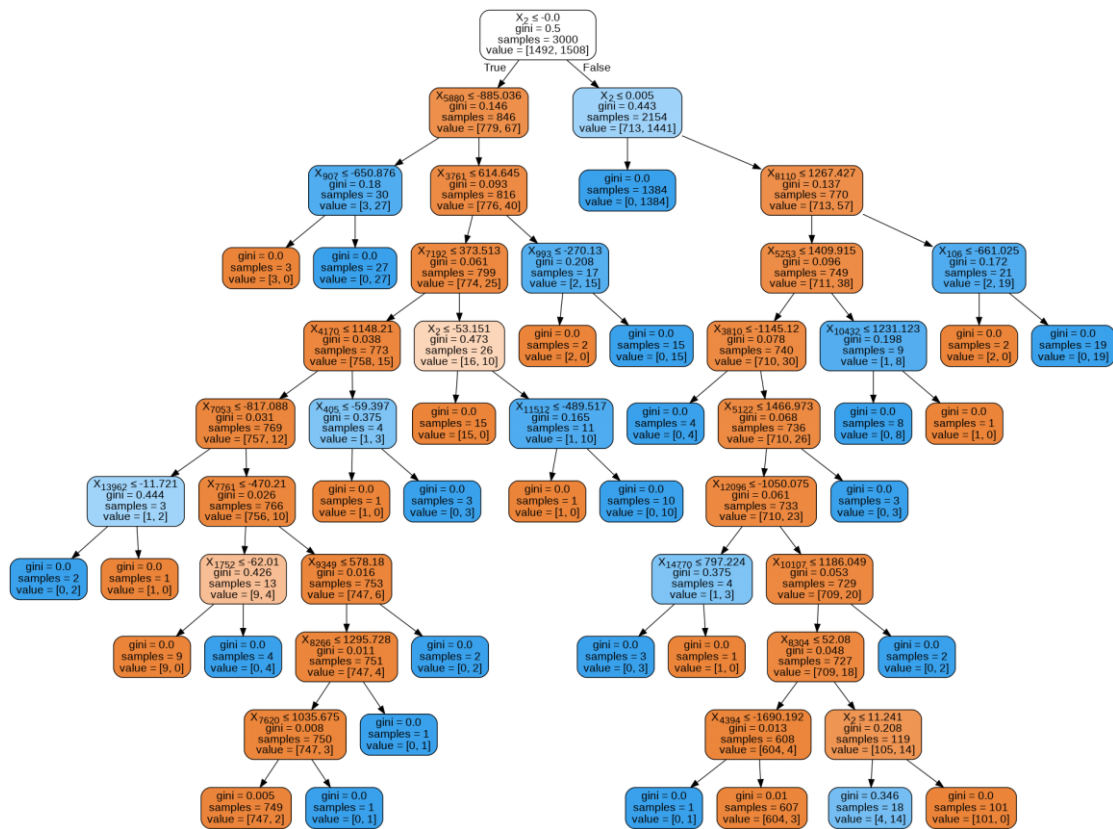
- 20만 개의 데이터셋으로 구성되어있는 chunk1(noise), chunk2(local_earthquake) 를 데이터셋으로 사용.
- h5py 패키지를 이용하여 chunk1, chunk2 의 엄청난 크기의 hdf5 파일을 전처리.
- csv 파일의 'trace_name' 인덱스를 이용하여 hdf5파일 데이터를 뽑아내고 numpy 를 이용하여 6000개의 window로 나누었다.
- 지진 signal 데이터는 p,s label을 1로 labeling한다.
- 노이즈 데이터는 p,s label을 0으로 labeling한다.
- pickle 패키지를 이용하여 머신러닝에 용이하게 사용할 수 있도록 데이터를 저장하고 가져온다.

3-1-2) Decision Tree model 학습

1) Data split

- 1만개의 sample을 가져와 train data 3 : test data 7 비율로 나누어 model을 학습시킨다.

2) Tree 시각화



[Fig3. Decision Tree 시각화]

- sklearn의 export_graphviz를 이용하여 Decision Tree를 시각화하였다.
- overfitting 방지하기 위해 max_depth는 10으로 설정하였다.

3) 분류결과

Train_Accuracy: 0.997

예측 정확도: 0.96

Accuracy: 0.96

Recall: 0.96

Precision: 0.96

F1_score: 0.96

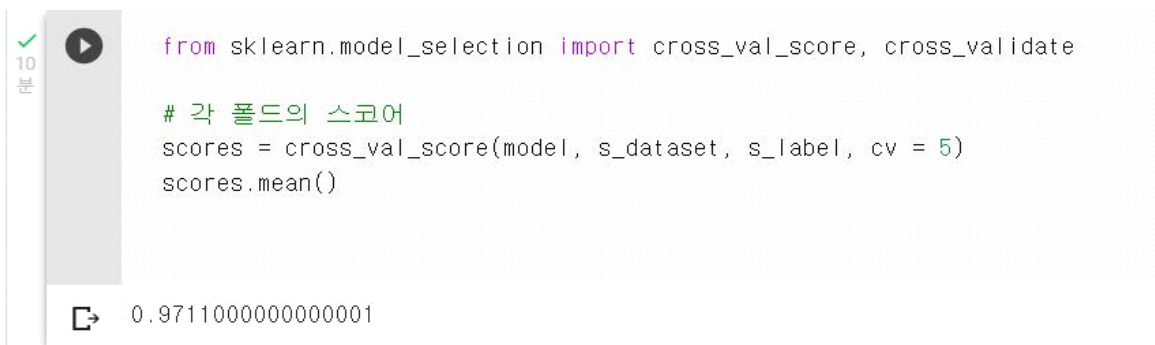
Confusion Matrix:

```
[[3356 131]
 [ 133 3380]]
```

[Fig4. Decision Tree Test결과]

- sklearn의 metric 라이브러리를 통해 훈련set에 대한 정확도, Test set에 대한 정확도, 재현율, 정밀도, F1_score를 평가하였다.
- 트리 분류 결과 단순 정확도가 0.96으로 굉장히 높게 나타났다. 분류 parameter가 1개의 클래스만 사용하였고 데이터 전처리시 1개의 클래스로만 label링 하였기 때문에 Tree 학습 결과가 좋은 것으로 판단된다.

4) 5-fold 검정



The screenshot shows a Jupyter Notebook cell with the following code:

```
from sklearn.model_selection import cross_val_score, cross_validate

# 각 폴드의 스코어
scores = cross_val_score(model, s_dataset, s_label, cv = 5)
scores.mean()
```

The output of the cell is displayed below the code:

```
0.9711000000000001
```

[Fig5. 5-fold 검정 결과]

- 검정 결과 10분 소요 검정 평균값은 0.97로 학습 데이터에 overfitting은 나타나지 않는것으로 보인다.

3-2. Model 2 (Customized CNN Model)

3-2-1) CNN 모델 사용 배경

- 3채널의 지진 Signal Data를 학습하기 위해 flatten시켜 P파/S파 관련 signal의 Feature를 극대화 시켜 학습하기 위해 도입하였다.
- convolution layer와 max pooling을 통해 지진 signal의 특징을 극대화하여 학습하면 P파와 S파 signal탐지를 할 수 있을 것으로 판단하였음

3-2-2) Model 학습과정

1. 'Test' 파일에는 비지진 데이터인 chunk1에서 1만개를 임의로 뽑고, 지진 데이터 중 chunk2에서 1만개를 임의로 뽑아 섞은 후 저장해둔 전처리 데이터가 저장되어 있다. 그 데이터를 pickle 파일 형식으로 읽어와 그 중에 2000개의 데이터만을 끌어온 뒤, E, N, Z축의 지진 데이터인 s_dataset과(sample dataset) 해당 데이터가 지진인지 비지진인지를 판별해주는 s_label(sample label)을 분리해 저장해준다. dataset 인덱스에서 p파의 존재 유무를 의미하는 p_label 데이터에 따라 해당 데이터가 지진인지 아닌지를 의미하기 때문에, p_label을 label 변수에 저장한다.

```
## Data split
sample_data = random.sample(data,2000)
s_dataset=[]
s_label=[]

for i in range(len(sample_data)):
    s_dataset.append(sample_data[i]['data'])
    s_label.append(sample_data[i]['p_label'])

s_dataset=np.array(s_dataset)
s_dataset=s_dataset.reshape(2000,18000,1)

print(s_dataset)

X_train, X_test, Y_train, Y_test = train_test_split(s_dataset, s_label, test_size=0.4, random_state=321)

X_train = np.array(X_train)
X_test = np.array(X_test)
Y_train = np.array(Y_train)
Y_test = np.array(Y_test)

print('X_train의 크기(shape) :', X_train.shape)
print('X_test의 크기(shape) :', X_test.shape)
print('Y_train의 크기(shape) :', Y_train.shape)
print('Y_test의 크기(shape) :', Y_test.shape)
```

[Fig 6. CNN model 학습 및 테스트 데이터 Split]

2. 하지만, 1D Convolution을 사용하기 위해서는 E, N, Z축으로 3D의 지진 데이터를 담은 데이터를 1D로 변환해주는 작업이 필요하므로, reshape 해주었다.

```
X_train의 크기(shape) : (1200, 18000, 1)
X_test의 크기(shape) : (800, 18000, 1)
Y_train의 크기(shape) : (1200,)
Y_test의 크기(shape) : (800,)
```

[Fig 7. Dataset reshape 과정]

3. train_test_split으로 데이터를 나누고 model 테스트를 진행한다.

```
## model definition
model = Sequential()
model.add(Conv1D(64, 2, activation="relu", input_shape=(18000,1)))
model.add(Dense(16, activation="relu"))
model.add(MaxPooling1D())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(1, activation='sigmoid'))

model.compile(loss = 'binary_crossentropy',
              optimizer = "adam",
              metrics = ['accuracy'])
model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
=====		
conv1d (Conv1D)	(None, 17999, 64)	192
dense (Dense)	(None, 17999, 16)	1040
max_pooling1d (MaxPooling1D)	(None, 8999, 16)	0
dense_1 (Dense)	(None, 8999, 128)	2176
dropout (Dropout)	(None, 8999, 128)	0
dense_2 (Dense)	(None, 8999, 1)	129
=====		

Total params: 3,537

Trainable params: 3,537

Non-trainable params: 0

[Fig8. Model2 Summary]

3-2-3) Model 2 결과

```
• pred = model.predict(X_train)
  pred_y = pred.argmax(axis=-1)

acc = model.evaluate(X_test, Y_test)
print("Loss:", acc[0], " Accuracy:", acc[1])

25/25 [=====] - 3s 103ms/step - loss: 0.6138 - accuracy: 0.6779
Loss: 0.6137675642967224 Accuracy: 0.6778897047042847
```

[Fig 9. Model2 Loss 및 Accuracy]

- Loss: 약 0.6137
- Accuracy: 0.6779

3-2-4) Model 2 결론

- CNN을 이용하여 지진 구분 및 P파/S파 탐지 모델을 목표하였으나 1D layer로 구성된 학습 모델로 인해 Signal 학습이 부족한 것으로 보여 지진 구분 결과가 좋지 못하였고 signal에서 P파/S파 phase checking도 불가능하였다.

4. CBA Model(CNN+ BiLSTM+ ATTENTION)

4.1 모델 소개

- 본 모델은 full wavelength 지진파에서 지진 검출과 P/S파 검출을 동시에 학습하는 딥러닝 모델을 hierarchical attention 기법을 이용하여 설계되었고, 이를 통해 지진 탐지와 P/S파 검출하고 검출 결과를 시각화 해주는 모델이다.
- 단순 CNN으로 구성된 모델 보다 정확도를 향상 시켰다.
- EQTransformer와 비교하여 정확도를 조금 낮추고 모델 학습시간과 탐지 속도를 높인 모델이다.
- 이전 Model2에서 불가능한 phase checking 기능을 완성하였고 시각화도 가능한 모델이다.
- 자연어 처리 분야에서 모든 task에 최고의 성능을 내는 BiLSTM Hegemony 에서 영감을 받아, 기존 CNN 모델을 개선을 위해 ATTENTION이 적용된 Bidirectional LSTM 을 사용하게 되었다. 또한 지진파는 continuous 한 데이터로써 직전 정보와 현시점 정보와의 관계가 있다. 이 기법을 이용하여, 직전 정보를 현 시점 정보에 더해줌으로써 그라디언트가 효과적으로 흐를 수 있게 한다.
- encoder 부분에 3개의 decoder를 연결하여 earthquake detection, p-phase picking, s-phase picking 3가지 task를 동시에 수행 할 수 있다.

4.2 DataSet

- Stanford Earthquake datase(STEAD) 을 네트워크 훈련에 사용하였다.
- 총 120만개의 STEAD 데이터 중 noise 200개와 local_earthquake 1000개를 추출하여 총 1200 개의 데이터셋을 이용하였다.
- 딥러닝 모델에서 유용하게 사용되는 hdf5 파일과, 라벨링된 csv 파일로 각각 데이터를 저장하여 사용하였다.
- 모델 훈련 및 평가에 사용하기위해 훈련, 검증, 테스트데이터를 6 : 2 : 2 의 비율로 나누어 사용하였다.

4.3 목적

○ 지진 검출

- 지진 검출은 여러 지진 관측 센서들로부터 노이즈 값과 함께 수집된 많은 seismic signal로부터 지진인지 아닌지를 판단하는 task이다.

○ P / S파 검출 (Phase picking) task

- P/S파 검출은 지진이 발생한 지역을 추측하기 위해 지진파로부터 S/V파를 판단하는 task이다

○ 모델 설계 아이디어

- 이전 모델들은 지진과 검출과 P/S파 검출을 각기 다른 네트워크를 통해 수행을 했지만, 본 모델은 1개의 모델로 2가지 task를 한번에 수행한다.
- 그 이유는 실제로 사람이 일을 할 때, 전체 데이터를 보고 phase를 분석하여 어떤 부분이 P/S파인지 분석한다. 이 점을 착안하여, 인간이 실제로 수행하는 방식을 모방하는 딥러닝 모델을 만들었다. 본 딥러닝 모델은 encoder를 통해 전체 데이터로부터 context 정보를 얻은 후, decoder에서 세부 task를 진행하는 방식으로 모델을 설계하였다. 상세한 모델 구조는 아래와 같다.

4.4 모델 구조

- 모델은 하나의 encoder와 세 개의 decoder로 구성되어 있다. encoder는 전체 인풋에서 지진과 관련된 feature를 강조시키는 역할을 하고, decoder는 지진 검출, P파 검출, S파 검출 등 특정 task를 수행한다. 네트워크 구조 설계는 인간이 해당 일을 수행하는 것을 모방하여 설계하였고, 설계한 네트워크의 parameter 최적화 작업은 계속 실험을 하면서 최적화해 나갔다.

○ One deep encoder

- encoder는 시간에 따른 seismic signal을 입력으로 받아, high-level representation과 context 정보를 생성해낸다. 들어오는 인풋의 길이가 길어지면 메모리가 많이 필요하므로 Conv 1D와 maxpooling을 이용하여 down sampling 해준다. 그리고 이렇게 down sampling 된 feature는 Res CNN과 Bidirectional LSTM을 통해 transform된 이후, encoder의 마지막에 이치한 global attention에 의해 지진과 연관된 feature가 커지게 된다

○ Three decoders

- 먼저 첫번째 decoder는 encoder의 입력 값을 받아, 지진 발생확률을 계산한다. 해당 decoder는 Conv 1D를 이용하여 down sampling된 데이터를 up sampling하고, sigmoid를 통해 결과 값을 도출한다.
- 나머지 2가지 decoder는 encoder의 입력 값을 받아, P파와 S파의 위치를 피킹한다. 해당 decoder는 Conv1D를 이용하여 up sampling하고, sigmoid를 통해 결과를 도출한다.

○ 그리고 모델의 각 block에 residual connection network-in-network 기법을 사용하여, 더 깊은 네트워크를 쌓을 수 있었다.

4.5 학습 방식

○ 데이터셋

=> STEAD(Stanford Earthquake Dataset) 사용: 1984년 1월 ~ 2018년 8월까지 미국과 유럽 등지에서 발생한 지진데이터 (1 Noise data + 5 Local Earthquakes data)

<https://rebrand.ly/chunk1> (chunk1 ~ 14.6 GB) Noise

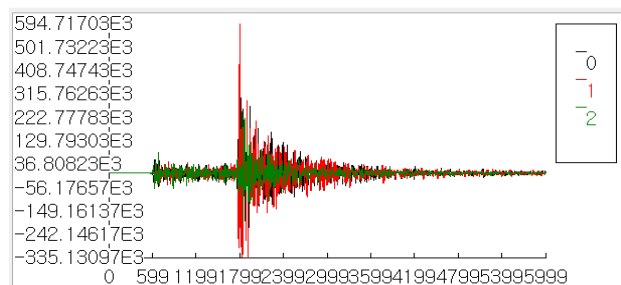
<https://rebrand.ly/chunk2> (chunk2 ~ 13.7 GB) Local Earthquakes

<https://rebrand.ly/chunk3> (chunk3 ~ 13.7 GB) Local Earthquakes

<https://rebrand.ly/chunk4> (chunk4 ~ 13.7 GB) Local Earthquakes

<https://rebrand.ly/chunk5> (chunk5 ~ 13.7 GB) Local Earthquakes

<https://rebrand.ly/chunk6> (chunk6 ~ 15.7 GB) Local Earthquakes



	A	B	C	D	E	F	G	H	I	J	K	L
1	network_c	receiver_c	receiver_ty	receiver_la	receiver_lc	receiver_el	p_arrival_s	p_status	p_weight	p_travel_s	s_arrival_s	s_status
2	TA	109C	BH	32.8889	-117.105	150	700	manual	0.5	17.08	1894	manual
3	TA	109C	BH	32.8889	-117.105	150	600	manual	0.5	16.88	1763	manual
4	TA	109C	BH	32.8889	-117.105	150	500	manual	0.5	17.26	1678	manual
5	TA	109C	BH	32.8889	-117.105	150	900	manual	0.5	17.28	2086	manual
6	TA	109C	BH	32.8889	-117.105	150	700	manual	0.5	18.14	1897	manual
7	IA	109C	BH	32.8889	-117.105	150	700	manual	0.5	18.02	1953	manual
8	TA	109C	BH	32.8889	-117.105	150	900	manual	0.5	17.87	2184	manual
9	TA	109C	BH	32.8889	-117.105	150	800	manual	0.5	18	2038	manual
10	TA	109C	BH	32.8889	-117.105	150	800	manual	0.5	17.98	2062	manual
11	TA	109C	BH	32.8889	-117.105	150	600	manual	0.5	17.18	1777	manual
12	TA	109C	BH	32.8889	-117.105	150	500	manual	0.5	12.9	1416	manual

[Fig
9.

학습 Dataset 파라미터와 신호 형태]

4.6 학습 과정

● 초기값 설정

- Conv / BiLSTM : Xavier normal initializer
- Bias: 0
- Learning rate: NADAM optimizer(0.001)

● 학습 환경

- Colab 이용 720개의 훈련 데이터를 사용하였을때 5시간 (100 epoch)
- validation loss 가 훈련중 감소하지 않으면 훈련 종료.

● 데이터 전처리

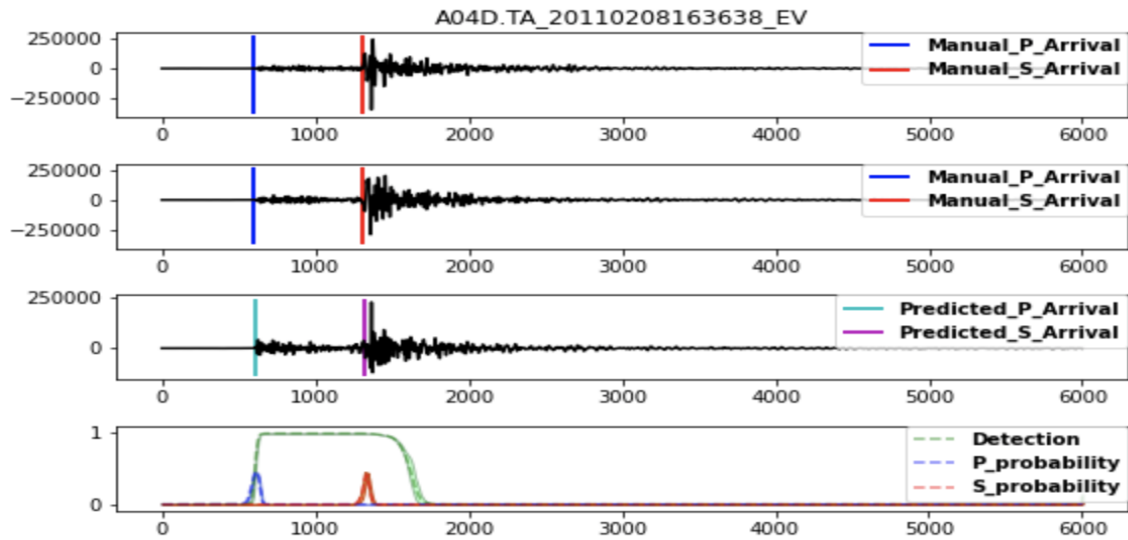
- Data augmentation 발생 -> Gaussian 노이즈 덧씌움, random shift, randomly adding gap, randomly dropping channels 방식 등으로 학습 데이터 Augmentation.

● 모델 훈련

- 위에서 언급한 encoder 와 decoder 를 이용하여 데이터를 훈련.

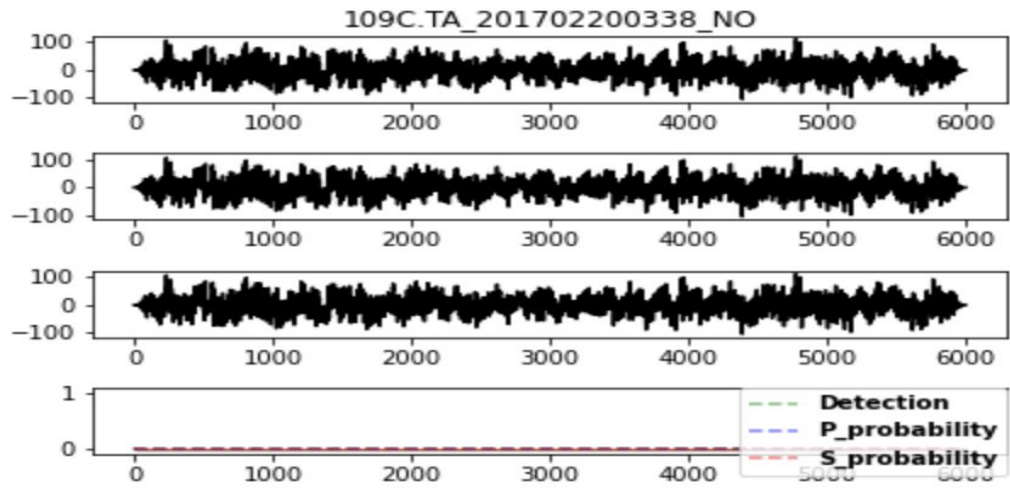
4. 결과

5.1 Signal 탐지 결과 시각화



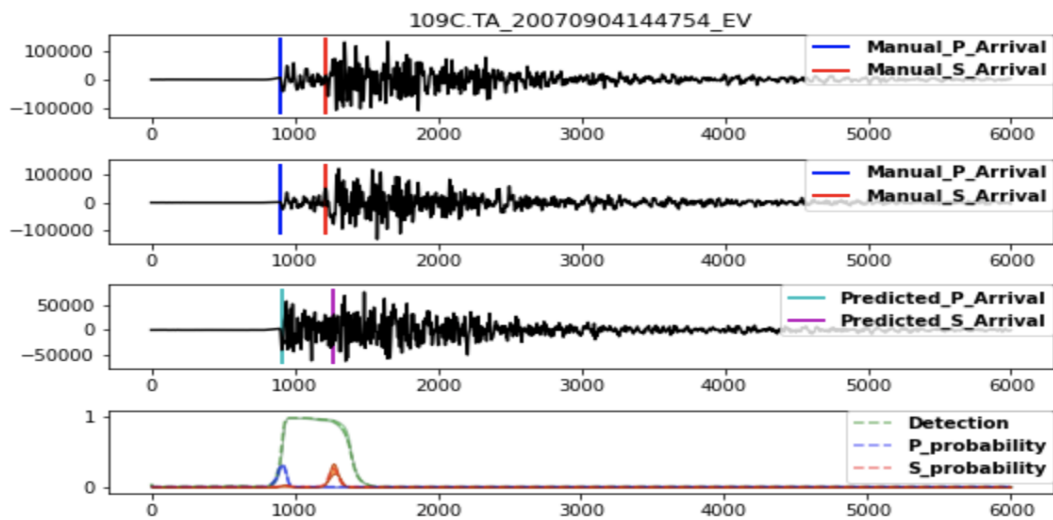
[Fig 10. Area A 탐지 결과]

- 탐지한 P파/S파의 phase checking이 실제 P파/S파 도착시간과 일치한 결과를 나타낸다.



[Fig 11. noise signal 탐지 결과]

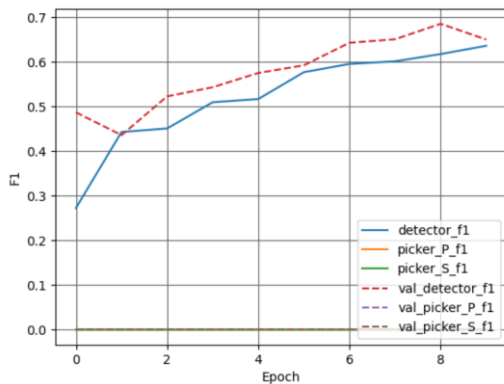
- Noise signal 경우 P,S과 Detection이 발생하지 않은 것을 볼 수 있다.



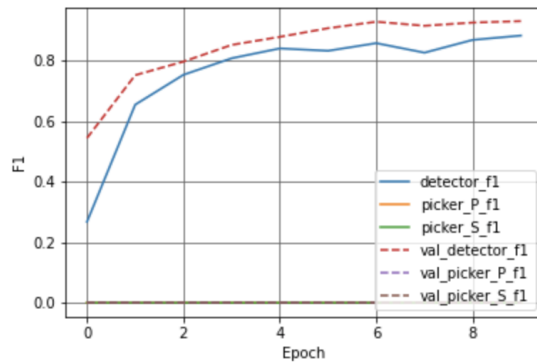
[Fig 12. Area C 탐지 결과]

- 또다른 Area C에 대한 p과와 S과 phase picking 결과이다,

5.2 Simple CNN Model(Model2)과 F1 Score 비교



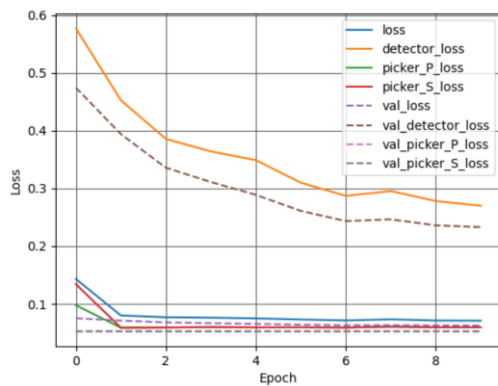
[Fig 13. Simple CNN Model(model2)-F1 Score]



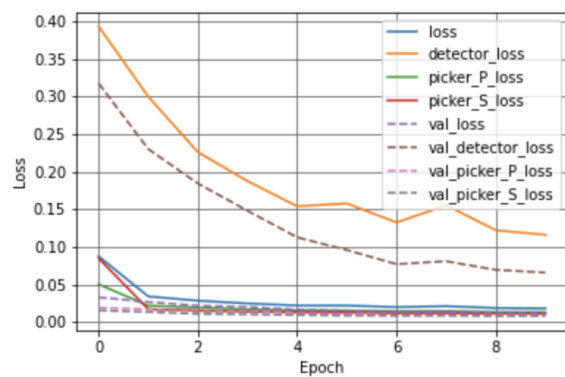
[Fig 14. CBA Model-F1 Score]

- (Model2 F1 Score: 0.69) 대비 (CBA Model F1 Score: 0.88) 로 F1 score가 향상한 것을 볼 수 있다.
- CBA Model은 epoch 0~2에서 f1-score가 급격하게 상승하였다.

5.3 Model2와 CBA Model Loss 비교



[Fig 15. Model2-Loss]



[Fig 16. CBA Model - Loss]

- Model2 Loss:0.05로 수렴 CBA Model Loss: 0.025로 수렴한다.
- CBA model의 Loss 값이 빠르게 줄어드는 것으로 보아 훈련 데이터로부터 가중치 매개변수 최적값을 빠르게 찾아내어 fitting 결과가 우수한 것으로 나타난다.

5.4 Model별 소요 시간 비교

5.4.1) Model2와 탐지 소요 시간 비교

(비교 환경: Test Data set : 총 **1200개** Train:Test:valid= **6:2:2** 비율)

기존 CNN만 사용한 모델 소요시간 : **26.85s**

CBA Model(BILSTM+Attention 기법 추가 적용) 모델: **30.58s**

⇒ Layer가 적은 CNN모델이 탐지 시간이 적게 걸렸으나 앞서 비교한 F1 score와 Loss에서 정확도가 현저 히 낮아 CBA Model이 우수한 것으로 나타난다.

5.4.2) EQTransformer와 비교

(테스트 환경: 1200 개의 test data, colab에서 구동)

1) 소요시간

EQTrasnformer: **38.9s**

CBA Model: **30.58s**

2) 탐지 정확도(precision)

EQTransformer: **0.99**

CBA Model: **0.92**

3) F1 score(1 최대)

EQTransformer: **1**

CBA Model: **0.88**

5.5) Model 훈련시 소요 리소스 비교

- EQTransformer : 450k의 continuous 데이터를 이용(noise + 지진)하여 4개의 parallel Tesla-V100 GPUs 로 12 epoch 로 구성시 **89Hours** 소요해 Model 학습
- CBA Model: 20k 개의 Signal 데이터를 이용하여 colab환경에서 GPU사용하여 10epoch로 학습시 **30Min** 소요해 Model 학습

6. 결론

chunk1에서 10000개, chunk2에서 10000개를 랜덤 추출하고 shuffle하여 만들어진 데이터셋을 이용하였다.

처음에 Decision Tree 기법을 사용한 모델을 이용하여 학습과 테스트를 진행한 결과 단순 정확도가 96%로 굉장히 높게 나타났다. 분류 parameter를 1개의 클래스만 사용하였고 데이터 전처리시 1개의 클래스를 label링 하였기 때문에 분류 Model의 정확도가 높게 나온것으로 보인다. 단순히 이미 발생한 Signal에서 지진과 Noise 분류에는 효과적이지만 P/S파 탐지하는데는 부족하여 조금 더 복잡한 Model 구현이 필요해 보인다.

두 번째로 만들었던 자체적으로 1D Convolution과 Maxpooling 등의 CNN 기법을 사용한 모델은 정확도가 68%로 좋지 않은 결과를 얻을 수 있었다. 이는, 우리가 해결하고자 하는 문제가 단순 CNN 기법만으로 해결할 수 있는 것이 아니라는 것을 알 수 있다.

마지막 모델인 CNN + Bidirectional LSTM + Attention 모델은 기존의 CNN 모델에서 시간의 Sequence에 따른 변화를 감지하는 양방향 LSTM 기법을 추가하니 88% F1 Score로 좀더 만족스러운 결과를 얻을 수 있었다.

하지만, 기존의 EQTransformer 모델의 99%라는 정확도에는 11% 모자라는 수준이다.

Model2 대비 소요시간은 증가하였으나 정확도와 Loss에서 성능이 더 좋았고 EQTransformer 대비 정확도는 0.07정도 부족하나 훈련시 필요한 리소스와 탐지 속도에서 앞서 일반 PC에서 가벼운 model로 사용 가능 할 것으로 보인다.

7. 향후 과제

- 보다 많은 Data 학습으로 CBA model의 정확도 향상
- Model의 학습 과정 및 테스트 과정 시각화를 도울 Tensorboard 라이브러리 추가
- 간단한 파라미터를 가진 Data도 분석할 수 있도록 모델 학습을 위한 인코더 개선

8. 수행 체계

8.1 구성원별 역할

이름	역할 분담
김장환	<ul style="list-style-type: none">과제 도중 사용될 딥러닝, 인공지능에 대한 지식 숙지 및 팀원에게 공유P/S파를 탐지하여 그것을 토대로 지진을 탐지하는 모델을 구축한다.구축된 새로운 모델을 다 같이 테스트해본다.각자 자체적인 CNN모델 구축 (지진파Detection)
전인혁	<ul style="list-style-type: none">연속 지진 데이터를 찾아 다운로드하여 테스트 데이터를 마련한다.P파 탐지를 위한 새로운 모델을 구축한다.구축된 새로운 모델을 다 같이 테스트해본다.각자 자체적인 CNN모델 구축 (지진파Detection)
나건혁	<ul style="list-style-type: none">과제를 수행 간의 필요한 개발 환경을 설정한다.S파 탐지 모델 구축구축된 새로운 모델을 다 같이 테스트해본다.각자 자체적인 CNN모델 구축 (지진파Detection)

8.2 개발 일정

5월	6월					7월				8월				9월												
4주	1주	2주	3주	4주	5주	1주	2주	3주	4주	1주	2주	3주	4주	1주	2주	3주	4주	5주								
AI 및 관련지식 공부																										
개발 환경 구축																										
기존 모델 테스트																										
중간 보고서																										
개선 모델 구현																										
P파 탐지 테스트 및 디버깅																										
노이즈 와 신호구분 테스트 및 디버깅																										
기존 데이터셋 이용하여 모델 테스트																										
개발 결과물 테스트																										
최종 발표/보고서																										

9. 참고 문헌

[1] Earthquake Detection and P-Wave Arrival Time Picking Using Capsule Neural Network. by Saad, Omar M.; Chen, Yangkang

IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING; JUL 2021, 59 7, p6234-p6243, 10p.

[2] M.Rhanoui, M. Mikram, Siham Yousfi, Soukaina Barzali(2019). A CNN-BiLSTM Model for Document-Level Sentiment Analysis.

[3] DLEP: A Deep Learning Model for Earthquake Prediction. by Li, Rui; Lu, Xiaobo; Li, Shuowei; Yang, Haipeng; Qiu, Jianfeng; Zhang, Lei

2020 International Joint Conference on Neural Networks (IJCNN) Neural Networks (IJCNN), 2020 International Joint Conference on. :1-8 Jul, 2020

[4] Schuster, Mike, and Kuldip K. Paliwal. "Bidirectional recurrent neural networks." IEEE Transactions on Signal Processing 45.11 (1997): 2673-2681.

[5] STanford EArthquake Dataset (STEAD): A Global Data Set of Seismic Signals for AI S. MOSTAFA MOUSAVI1, YIXIAO SHENG1, WEIQIANG ZHU1, and GREGORY C. BEROZA