1.

1. LSTM 모델을 완성하는 단계로서, K-fold cross validation 을 사용하면 인공지능 모델의 성능이 그렇지 않는 모델과 비교에서 더 신뢰할 수 있다고 한다. K-fold cross validation 이 무엇인지 설명하라

K-fold cross validation을 사용하는 가장 큰 이유는 데이터 수가 적기 때문에 사용한다. 데이터의 개수가 적으면 데이터 부족으로 인한 underfitting이 일어날 수 있기 때문에 이를 방지해준다. 단점은 그냥 Training set / Test set 을 통해 진행하는 일반적인 학습법에 비해 시간 소요가 크다

1. Training Set과 Test Set을 나눈다.

2. Training을 K개의 fold로 나눈다.

3. 한 개의 Fold에 있는 데이터를 다시 K개로 쪼갠다음, K-1개는 Training Data, 마지막 한개는 Validation Data set으로 지정한다

4. 모델을 생성하고 예측을 진행하여, 이에 대한 에러값을 추출한다

5. 다음 Fold에서는 Validation셋을 바꿔서 지정하고, 이전 Fold에서 Validatoin 역할을 했던 Set은 다시 Training set으로 활용한다

6. 이를 K번 반복한다.

7. 각각의 Fold의 시도에서 기록된 Error를 바탕(에러들의 평균)으로 최적의 모델(조건)을 찾는다

8. 해당 모델을 처음에 분할하였던 Test set을 활용하여 평가한다

2.

2. 최종 인공지능 모델을 구축한 후, 나중에 LSTM 모델을 사용하기 위해서는 메모리에 저장해 둘 필요가 있다. 저장 방법은 어떤 것이 있는지 설명하라

1. model.save() 메서드를 사용하여 전체 모델을 저장할 수 있습니다.

2. with open(‘architecture.json’,‘wt’) as json\_file:

json\_file.write(architecture)을 통해 아키텍처를 저장할 수 있습니다.

json\_file = open(‘architecture.json’,‘rt’)

architecture = json\_file.read()를 통해 아키텍처를 불러올 수 있습니다.

model.save\_weights(“filename.h5”);

model = ...

model.load\_weights(“filename.h5”);

를 통해 가중치를 저장,불러올 수 있습니다.

3. 저장할 수 있는 데이터포멧은 json yaml 등이 있습니다.

3.

3. LSTM 모델의 성능결과(혹은 learning curve)를 분석할 때, 모델이 overfitting 과 underfitting 이라고 판단하는 기준에 대해서 설명하라.

Overfitting -epoch가 일정 수준을 넘어감에 따라 모델이 training dataset에 overfitting 되면 오히려 validation loss가 증가하게 됩니다. 즉, validation loss가 감소하다가 증가하는 구간에서 Overfitting 이 발생했다고 말합니다. 분산은 작지만 데이터가 전체적으로 target으로부터 떨어져 있어 편향이 크다면 overfitting

Underfitting - 모델의 학습이 수행되면서 train loss와 validation loss가 함께 감소하며 학습이 될수록 모델의 성능이 좋아지는데에도 불구하고 학습을 멈춘다면 Underfitting이 발생했다고 말합니다.

4.

4.모델의 성능이 overfitting 혹은 underfitting 로 확인되었을 때, 이를 해결하기 위한 방법은 무엇인가?

Overfit: 모델이 학습 데이터에 비해 과하게 복잡하지 않도록, hidden layer 크기를 줄이거나 layer 개수를 줄이는 등 모델을 간단하게 만듭니다.

Dropout: 학습을 할 때 일부 뉴런을 끄고 학습합니다.

데이터를 정규화 시켜줍니다.

학습 데이터가 편향되어 있지 않은지 확인할 필요가 있습니다.

validation loss가 가장 낮은 지점인 최적 epoch에서의 모델을 저장해준다.(한두번 epoch에서는 감소안할수있기떄문에 최소 3번이상으로 설정)

Underfit

학습 반복 횟수를 늘려줌

데이터의 특성에 비해 모델이 너무 간단한 경우 언더핏이 발생할 수 있음 따라서 모델 파라미터가 더 많은 강력한 모델을 구축합니다.

데이터의 양을 늘려준다.

모델의 제약을 줄입니다.

5.

5. LSTM 모델을 구축할 때 overfitting 문제를 해결하고 정규화된 모델을 만들기 위해서는 Dropout 을 사용한다. Dropout 에 대해서 설명하라.

Dropout은 딥러닝 모델의 과적합을 방지하기 위해 사용되는 정규화 기법 중 하나입니다. Dropout은 훈련 과정에서 특정 뉴런을 무작위로 제거하여 모델이 특정 뉴런에 과도하게 편향되지 않도록 합니다. Dropout을 사용하면 모델이 데이터의 특정 부분에 의존하지 않고 더 일반적인 패턴을 학습할 수 있습니다. Dropout은 실제 값을 예측할때는 사용하지 않는다.

예시) A라는 사람이 어떤 여성의 총 10가지의 특징을 보고 좋아했다고 한다면, 학습을 할 때 4가지 정도를 날려버린다는 것이다. 그게 얼굴 모형이 될수도 있고, 코 모양이 될 수도 있고 되도록이면 안 날리는 것이 중요하겠지만, 제대로 된 학습 모형을 만들기 위해서 오버핏한 데이터의 일부를 날려버린다.

6.

6. LSTM 아키텍처가 시퀀스 예측문제를 잘 해결할 수 없어서, 성능을 개선하려고 한다. 성능을 개선하기 위해서는 LSTM 아키텍처를 어떻게 변경할 것인지를 설명하라.

모델 기술에 대한 다양한 데이터 값 스케일링 체계의 효과를 평가합니다. 첫 번째 숨겨진 레이어 및/또는 출력 레이어에서 활성화 함수를 업데이트하여 입력으로 제공되거나 출력으로 예측되는 값의 범위를 처리해야 합니다. 시도해 볼 수 있는 몇 가지 체계는 다음과 같습니다.

값을 정규화합니다.

값을 표준화합니다.

모델 기술에 대한 다양한 값 인코딩의 효과를 평가합니다. 문자나 단어와 같은 일련의 레이블은 종종 정수로 인코딩되고 하나는 핫 인코딩됩니다. 이것이 시퀀스 예측 문제에 대한 가장 숙련된 접근 방식이라는 가정을 테스트합니다. 시도해 볼 몇 가지 인코딩 체계는 다음과 같습니다. 정수값,실제값,원핫인코딩

시계열과 같은 일련의 실제 값으로 작업할 때 계열을 정지 상태로 만드는 것이 좋습니다.

입력 시퀀스 길이의 선택은 문제 영역에 따라 다를 수 있지만 때로는 LSTM에 대한 문제를 프레이밍하는 일부로 선택할 수도 있습니다. 모델의 기술에 대해 다른 입력 시퀀스 길이를 사용하는 효과를 평가합니다. 입력 시퀀스의 길이는 가중치를 업데이트할 때 오류 기울기를 추정하는 데 사용되는 시간을 통한 역전파에도 영향을 미칩니다. 모델이 학습하는 속도와 학습 내용에 영향을 미칠 수 있습니다.

시계열과 같은 일련의 실제 값으로 작업할 때 계열을 정지 상태로 만드는 것이 좋습니다.

trends제거 Seasonality 제거 Variance 제거

Dropout: 학습을 할 때 일부 뉴런을 끄고 학습합니다.

7.

7. LSTM 모델을 사용하여 생성형 AI 를 개발하려고 한다. 생성형 모델과 예측모델의 주된 차이점이 무엇인지 설명하라.

생성형 모델: 생성형 모델은 주어진 데이터의 분포나 패턴을 학습하여 새로운 데이터를 생성하는 데 초점을 맞춥니다. 모델은 기존 데이터에서 특정 패턴을 학습하고, 그 패턴을 기반으로 새로운 데이터를 생성합니다. LSTM 모델의 출력을 입력으로 사용하여 새로운 시퀀스를 생성하는 방식으로 구성됩니다.

예측 모델: 예측 모델은 입력 데이터에 대한 정확한 결과 값을 예측하는 것을 목표로 합니다. 이 모델은 주어진 입력 시퀀스를 기반으로 다음 값을 예측하고, 예측된 결과와 실제 결과 간의 오차를 최소화하려고 합니다. 입력 시퀀스를 처리하고 LSTM 모델의 출력을 활용하여 다음 값을 예측하는 방식으로 구성됩니다.

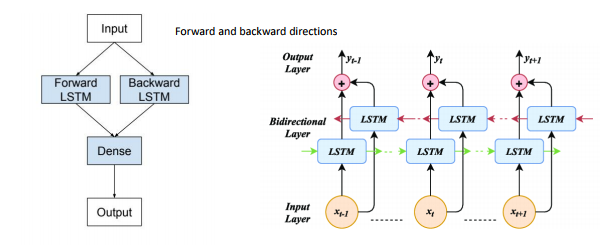
생성형: 위키피디아 기사 작성, 컴퓨터 코드작성, 기사 헤드라인 작성, 작곡

예측모델: 내일 날씨 예측, 고객 만족도 예측 등

8.

8. Bidirectional LSTM (BiLSTM)의 주된 개념은 무엇인가? BiLSTM 아키텍처를 블록 다이어그램으로 그려라.

입력 시퀀스를 양방향으로 처리하는 LSTM 아키텍처입니다. 기존의 LSTM은 입력 시퀀스를 단방향으로 처리하여 이전 정보만 참조합니다. 반면, BiLSTM은 입력 시퀀스를 양방향으로 처리하여 이전 정보와 이후 정보를 모두 활용합니다. 기존 lstm은 단방향으로 참조하였기 때문에 이전에 학습한것에 대해 gradiant vanishing 현상이 일어날 수 있다 하지만 bi를 쓰면 이 문제를 해결할 수 있다. forward와 backword 둘다 사용



9.

9. BiLSTM 이 일반 LSTM 보다 우수한 성능을 얻을 수 있는 응용 사례를 들고 왜 BiLSTM 이 우수한지를 설명하라.

bi를 사용하면 빨리 예측하고 확률을 높일 수 있다. 환자의 심박수를 몇일 측정 했더니 어떠한 패턴이 나오면 사망 bi 사용하면 바로 빠르게 예측가능 BiLSTM은 양방향으로 입력 시퀀스를 처리하므로, 단방향 LSTM보다 좀 더 넓은 맥락을 고려할 수 있습니다. 이는 문장이나 문서와 같은 시퀀스 데이터에서 양방향적인 의미와 문맥을 파악하는 데 도움이 됩니다.

10.

10. keras 라이브러리를 이용하여 BiLSTM 모델을 구축하는 파이썬 코드를 아래의 아키텍처 조건에 맞도록 작성하라. 메모리 셀 수 = 50, 샘플수=10, 입력데이터 개수=20, 특징(feature 혹은 데이터 차원)=1, 출력데이터=1, 활성함수= sigmoid, 손실함수= binary\_crossentropy, 최적화 알고리즘= adam, 싱글 히든 레이어(single hidden layer)를 적용하라.

import numpy as np

from keras.models import Sequential

from keras.layers import LSTM, Dense, Bidirectional

#입력 데이터

X\_train = np.random.randn(10, 20, 1)# shape: (샘플수, 입력데이터 개수, 특징)

y\_train = np.random.randint(2, size=(10, 1)) # shape: (샘플수, 출력데이터)

model = Sequential()

model.add(Bidirectional(LSTM(50), input\_shape=(20, 1))) # 메모리 셀 수, 입력데이터 개수, 특징

model.add(Dense(1, activation='sigmoid')) # 출력데이터

#모델 컴파일

model.compile(loss='binary\_crossentropy', optimizer='adam') #손실함수, 최적화 알고리즘

#모델 학습

model.fit(X\_train, y\_train, epochs=10, batch\_size=32) # epoch, batch size

predictions = model.predict(X\_train)

print(predictions)

11. Encoder-Decoder LSTM 은 종종 “시퀀스 임베딩”으로 알려져 있다. Encoder-Decoder LSTM 의 핵심 개념을 설명하라.

Encoder : 각 단어를 받아서 최종적으로 고정길이의 문맥백터를 만듬

입력 시계열 데이터에서 각 단어들간의 문맥을 포함함

sequence embedding을 입력으로 받아서 Context vector

엔코더의 입력이됨

입력 데이터를 seq2seq모델로 표현

Decoder : 고정길이로 암호화된 벡터를 풀어서 예측 시계열 데이터를 출력함 자연어처리용 으로 활용 입력과 출력의 데이터 길이의 변화는 벡터표현으로 처리 가능

• 기계번역(machine translation)

• 프로그램 실행학습 (learning to execute)

• 이미지 설명 (image captioning)

• 대화형 모델 (conversational model) (챗봇)

• 이동 분류 (movement classification)

12.

12. Encoder-Decoder LSTM 아키텍처를 블록 다이어그램으로 그려라.

