241122 순환신경망과 NLP

▼ 시계열 데이터와 순환신경망

시계열 데이터란?

- 시계열 데이터: 시간의 흐름에 따라 순차적으로 측정된 데이터
- 특징:
  - 일정 시간 간격으로 배치된 데이터들의 수열
  - 시간이 중요한 변수로 작용한다.
  - 시간의 흐름에 따른 변화를 관찰할 수 있음
  - 무작위로 섞어놓으면 의미없는 데이터가 됨
    - 위 특징에 따라 문자열, 동영상, 오디오 등도 시계열 데이터
- 기존모델+시계열?
  - 주식 가격, 시간 2개 변수만으로 특정 시점에서의 주식 가격 측정 → 제일 중요한 데이터?
    - → 그 시점에서의 과거 주식 가격, 미래 주식 가
  - 기존 모델은 시간축과 관계 없이 데이터를 random하게 처리하여 학습
    - → 관계를 볼 수 없음

RNN

- 임의의 길이를 가진 시퀀스 데이터나 시계열 데이터 분석이 가능한 신경망
- 특징:
  - 문장, 문서, 오디오 샘플 등을 입력으로 받기 가능
  - 자동 번역, 스피치 투 텍스트 등 자연어 처리에 유용
- 작동:
  - 순환 뉴런:
    - 입력을 받아 출력을 만들고 자신에게도 출력을 보내는 뉴런
    - 각 타임스텝마다 입력과 이전 타임스텝의 출력을 입력으로 받음
  - 순환 층:

        - 모든 뉴런이 타임 스텝 t마다 입력 벡터와 이전 타임스텝의 출력을 입력으로 받음
        - 입력과 출력이 벡터
        - 입력을 위한 가중치, 이전 타임스텝 출력을 위한 가중치
- 단점: 처음 입력값이 크게 흐려짐

BPTT

- 타임스텝으로 네트워크를 펼치고 역전파를 사용해 업데이트
- 비용 함수의 그레이디언트를 펼쳐진 네트워크를 따라 역방향으로 전파
- 비용 함수를 계산한 모든 출력에서 역방향으로 전파됨
- 각 타임스텝마다 같은 매개변수가 사용되기 때문에 모든 타임스텝에 걸쳐 합산됨

LSTM

- RNN의 단점을 극복하기 위해 나온 대표적인 모델
    - 장기 기억의 부재
    - 학습속도 저하 문제 (연산량 多)
    - gradient 소실
- Forget gate: 기억할 정보를 선별함
- cell state: 장기 상태라고도 불림
    - → RNN보다 장기기억력 ↑, gradient 소실 문제 개선
- + 학회스터디 깃허브 안에 자세한 설명이 있음!!
- + seq2seq, seq2vec, vec2seq
    - Sequence to Vector Network
        - sequence를 입력으로 받고 vector를 출력하는 네트워크
        - 예시:
            - 문장을 입력받고 감정을 분류하는 모델
            - 이미지 묘사 프롬프트를 입력받고 이미지를 출력하는 모델
    - Vector to Sequence Network
        - vector를 입력으로 받고 sequence를 출력하는 모델
        - 예시:

        - 이미지를 입력받고 서술을 출력하는 모델
        - 단일 단어를 입력받고 서술을 출력하는 모델
    - Sequence to Sequence Network
        - 입력과 출력 모두 sequence인 네트워크
        - 주식 데이터 같은 시계열 데이터 예측에 유용
        - 단점: 일반적인 네트워크의 형태로는 입력과 출력 사이즈가 달라짐. 대응불가.
- Encoder-Decoder 구조
    - NLP에서 seq2seq의 대부분이 이 구조
    - encoder 입력 → context vector 변환 → decoder 출력
    - 주로 LSTM과 같은 개선된 구조를 사용해 구현
    - 예시: 번역기, 텍스트 요약

▼ 자연어처리와 LLM

자연어 처리의 특징

- 자연어 처리가 어려운 이유:
    - 컴퓨터는 숫자로 정량화가 가능한 데이터만 처리 가능
    - 기존의 자연어 변환 방식은 단순한 규칙에 불과
        - 문자를 나타내지만 뜻을 나타내진 않음
    - 자연어를 제대로 처리하기 위해서는 자연어의 의미를 함축하게 만들어야 함
        - 여러 작업을 거칠 필요가 있음
        - 현재도 개선 중
- Tokenization (토큰화)
    - 정량화를 하기 전에, 자연어의 천문학적인 경우의 수를 '의미를 가진 단위'로 나눔
    - → 변환 가능한 형태로 만들어둠
    - 형태소 별로 나누는 방법이 일반적
        - 생성형 AI 서비스에서의 token이 가진 의미가 이것

자연어 전처리

- Word Embedding
  - 자연어 전처리에서 가장 어렵고 중요한 작업
  - 각 토큰을 의미가 있는 vector로 변환. 매우 고차원의 데이터.
  - 초기에는 원-핫 인코딩 → 문장 내 분포, 동시 언급...(Word2Vec, CBOW) → 최근에는 LLM 사용 방법론 등이 연구됨

순환신경망의 한계

- encoder-decoder 구조의 문제
  - context vector에 모든 정보 압축하니 정보가 손실. 입력이 길어지는 만큼 손실 발생
  - decoder에서 입력 sequence를 한번 더 살펴보며 추론하게 만들어야 함
  - 해당 출력 시점에서 중요한 입력 시점이 어딘지 정보 부여 필요

어텐션 알고리즘과 트랜스포머

- Attention Mechanism
  - seq2seq의 구조적인 문제를 개선하기 위해 연구된 대표적인 알고리즘
  - decoder에서 출력 단어를 예측하는 시점마다 encoder에서 전체 입력 문장을 다시 참고한다.
  - 골고루 참고하는 것이 아니라, 해당 시점에서 예측해야 할 단어와 연관이 있는 부분에 더 집중
  - 변수:
    - Q = Query: t 시점의 decoder 셀의 은닉 상태
    - K = Keys: 모든 시점의 encoder 셀의 은닉 상태들
    - V = Value: 모든 시점의 encoder 셀의 은닉 상태들
  - 주어진 쿼리에 대해서 모든 키와의 유사도를 각각 구하고, 키와 맵핑된 각 값에 반영
  - 유사도가 반영된 값을 보두 더하여 리턴
- Transformer
  - Attention mechanism만으로 구성한 딥러닝 모델
  - 현재 모든 LLM, 나아가 Multi Modal Model의 중추가 되는 알고리즘
  - 같은 파라미터 수 대비 RNN보다 빠른 속도

    - Q, K, V가 모두 같은 self attention을 사용
        - Q = Query: 입력 문장의 모든 단어 벡터들
        - K = Keys: 입력 문장의 모든 단어 벡터들
        - V = Value: 입력 문장의 모든 단어 벡터들
            - 같은 문장 내 모든 단어 쌍 사이의 의미적, 문법적 관계를 포착
            - multi-head attention은 최선의 결과를 내기 위해 어텐션을 여러 번 시행
    - 어떤 쿼리와 키가 연관이 크다면, 내적이 커지는 방향으로 학습
    - 내적이 크다는 건, 두 벡터가 공간 상 비슷하게 위치한다는 뜻
    - 토큰 사이의 관계를 반영해주는 방향으로 학습
    - 그 후 softmax 함수를 통해 확률값으로 변환

```
입력 행렬 X
쿼리에 해당하는 가중치 행렬 W-q
키에 해당하는 가중치 행렬 W-k
값에 해당하는 가중치 행렬 W-v
    X * W-q = Q
    X * W-k = K
    X * W-v = V
```

- Transformer 응용
    - 생성형 모델에서는 encoder-decoder 구조일 이유가 없음
    - decoder 부분에 바로 입력해서 출력을 받는 형식으로 구현 가능
        - 토큰을 하나씩 출력 → 출력한 토큰을 다시 입력 ← 과정을 반복
        - 최종 출력을 만들어내는 것이 생성형 모델의 작동 원리
- Transformer 기반 LLM
    - 장점:
        - 이전의 언어모델보다 월등히 뛰어난 성능
            - 많은 양의 텍스트로도 사용 가능
            - 파라미터 수를 늘리거나 좋은 훈련 데이터를 사용 → 다양한 task에 적용 가능

        - 다양한 훈련과 평가 기법이 연구됨
            - 자연어 생성 뿐만이 아니라 image, audio 등 다양한 분야에 접목
    - 단점:
        - 학습 및 튜닝에 고비용 소모
            - 막대한 파라미터 수로 이용에 어려움
        - 데이터 편향으로 인한 문제 야기
            - 확률 모델의 특성으로 인한 환각 현상
        - Interpretability의 부재

과제

수업 내용 정리

자신의 파일, prompt 또는 API를 사용해 LangChain code 실습