1. **Sequence Modleing**

우리들 모두는 시공간속에서 살고 있잖아요?   
그렇기 때문에 우리 생활속의 어떠한 문제를 해결할 때 시간 및 순서정보는 큰 영향을 줍니다.  
예를들어 주식이나 기상정보 음성인식 그리고 우리가 지금 공부하는 자연어 처리 분야에도 시간 및 순서 정보는 영향을 주게 됩니다.  
그러면 이제부터 이러한 순차적인 정보를 어떻게 학습할지에 대해서 설명하도록 하겠습니다.

1. **Sequence Model을 다루기 위해서는?**

그러면 sequence model 을 다루기 위해서는 어떠한것들이 필요할까요?  
첫째로 데이터가 순서에 대한 정보를 가지고 있어야합니다  
그다음 이전 sequence에 대한 내용이 다음 sequence에 전해져야합니다.  
순서 정보에 대한 느낌을 주기 위해 아래 같은 예를 들어 보았는데요  
어제 제가 ch1을 읽었다면, 오늘은 어디를 읽어야 할까요???   
네, 오늘은 ch2를 읽겠죠.. 이렇게 어제 ch1 읽은 정보를 통해 오늘 읽은 정보를 얻는 것이 sequence모델링 이라고 할 수 있습니다.

1. **State in Sequence**

이 Sequence의 핵심은 State인데요. state 는 사전적 의미로는 상태죠?  
그럼 Deeplearning에서의 state는 무엇일까요?  
현재의 정보를 담고 있는 존재  
이전의 정보 또한 가지고 있는 존재  
그리고 다음에게 이 모든 정보를 넘겨줄수 있는 존재가 state입니다.  
 State 를 변화순서대로 연결한 것을 Sequence라고 합니다.  
하지만 기존의 DNN에서는 데이터가 서로 독립적이기 때문에 stateful machine을 구현할 수가 없어 앞으로 배울 RNN이 등장하게 되었습니다.  
저는 Sequence modeling에서 state개념이 굉장히 중요하다고 생각해서 책엔 없지만 위 내용을 준비해 보았습니다

1. **RNN for Sequence Modeling**

순환신경망 RNN은 그림과 같이 input값 x와 직전의 은닉상태(hidden state)를 참조하여 현재의 상태 ht를 결정하는 작업을 여러 time-step에 걸쳐 수행합니다.

1. **값이 앞으로 전달되는 과정 Feed Forward**이전에도 말하고 책 그림에도 나와있듯 RNN은 각 time-step 별로 input값과 이전 hiden-state값을 통해 예측값을 반환합니다. 이렇게 얻어낸 예측값과 정답값을 비교해 손실 Loss를 계산 하게 됩니다.  
   아 ,여기서 주의할점은 RNN은 하나의 함수와 하나의 가중치를 계속해서 사용한다는 점입니다. 그리고 책에서도 나와있듯이 모든 파라미터가 실수인것과 과거에 활성화 함수로 하이퍼볼릭탄젠트가 쓰이는 이유에 대해서는 자주 언급되는 흥미로운 주제이니, 각자 한번쯤 알아 보시는것도 좋을 것 같습니다  
   **(**<https://curt-park.github.io/2017-04-03/why-is-lstm-strong-on-gradient-vanishing/>)
2. **BPTT(Backpropagation Trough Time)**RNN은 시간에 따라 펼쳐놓으면 구조가 MLP(Multi-Layer Perceptron)와 유사하기 때문에 BackPropagation 방법으로 계산이 가능합니다   
   이전 Backpropation과 다르게 뒤에 TT가 추가 되었는데요, 이름을 보았을때 시간에 따른 처리를 한다는 것을 한다는 것을 알 수 있을 것 같습니다.
3. **Vanishing Gradient**RNN 은 긴 시퀀스를 처리하는데 한계가 있는데요  
   예로 주요 단어들 사이에 여러 시간 스텝이 지난다면 앞의 정보를 잘 기억 못한다는 것입니다.  
   이러한 문제는 위에 말한 BPTT 과정 중에 발생하게 됩니다.   
   자세한 설명은 안하겠지만, ppt의 파란색으로 포커스한 값은 chain rule에 의해서 아래와 같은 식으로 나누어지게 됩니다.  
   문제는 우리가 선택한 활성화 함수가 tanh라는 점인데, 이 tanh의 미분값이 0~1사이라는 것을 감안한다면 층을 거칠수록 기울기의 크기는 작아질 수 밖에 없다는 것을 알 수 있습니다.  
   이러한 문제를 vanishing gradient라고   
   이러한 문제를 해결하기 위해 LSTM과 같은 RNN의 변형이 나오게 되었고   
   현재는 렐루나 레지듀얼 커넥션등을 통해 기울기 소실 문제를 해결하였다고 합니다.
4. **Multilayer RNN & Bi-direcional RNN**여러 개층을 갖는 RNN, 양방향 RNN  
   Attention이 도입되면서 사실상 쓸모없는 개념인 것
5. **자연어 처리에 RNN을 적용하는 사례**
6. **LSTM**LSTM, long short term memory가 어떻게 구현되는지 알아보도록 하겠습니다  
   lstm은 state를 표현하는 h와 함께 이전상태 기억을 표현하는 메모리셀 c를 가집니다.  
   state h가 변하더라도 잊지말아야할 정보를 cell state c에 담아둡니다  
   그러나 Memory에 정보를 무한히 담아둘수는 없으니 state h를 봐서 불필요한 정보는 버리도록합니다.
7. **Lstm의 메모리의 구조는 아래와 같습니다.  
   forget gate** : 어떤 기억을 보존할까?  
    이전 state와 새로운 입력 정보를 보아, 이전 기억을 얼마나 보존할지(잊어버릴지) 결정한다. 메모리의 정보를 f\_t가 0에 가까울수록 많이 잊어버리고, 1에 가까울수록 많이 보존한다  
   **input gate** : 새롭게 기억해야 할 것은?  
    이전 state와 새로운 입력 정보를 비교하여, 새롭게 추가된 정보가 얼마나 되는지, 그 정보를 얼마나 기억할지를 0~1 사이로 결정한다.  
   **cell state** : 메모리관리  
    잊어버릴 정보는 잊어버리고, 새롭게 기억해야 할 정보는 추가하여 재구성한다.  
   **output gate** : 새롭게 재구성된 기억으로부터 다음 state를 결정한다, 하지만 기억 중 모든 것이 state를 규정하는 것은 아니므로, o\_t를
8. 통해 state를 규정하는 요소들만 모아 본다  
     
   이러한 구조로 인해 lstm은 rnn에 비해 정확히 4배의 학습가능한 파라미터를 가지고 있습니다.
9. **GRU (gated recurrent unit)**gru는 lstm의 간소화 버전입니다. 기존 LSTM이 복잡한 모델인데 비해 GRU는 간단하면서도 성능이 비슷하다고 합니다.   
   하지만 모델에 따라 파라미터 셋팅을 다시 찾아내야 한다는 단점? 때문에 LSTM의 사용빈도가 더 높다고 합니다.