〈데이터분석과기계학습 12주차〉

인공신경망 (3) - RNN

인공지능융합공학부 데이터사이언스전공

곽찬희

오늘은!

- 순환신경망(Recurrent Neural Network, RNN) 과 시퀀스 데이터의 이해
- 최대한 수학은 줄이고 원리 이해에 집중할 수 있도록 구성했습니다.

통계적으로는

- Moving Average - 특정 기간 안의 평균으로 값을 추정해봅시다
- Exponential Smoothing - 시간에 따른 영향을 exponential로 표현
- ARIMA (Auto Regressive Integrated Moving Average)
- SARIMA (Seasonal ARIMA)
- Time Series Regression with Auto Correlation (error 항의 자기상관 허용)
- ARIMAX
- Prophet -> Facebook!!
- 기타 등등등…

Prophet만 잠깐

- g(t): 반복되지 않는 트렌드

C - Carrying Capacity, K - growth rate, M- Offset parameter

$$g(t) = \frac{C}{1+\exp(-k(t-m))}$$

- s(t): Seasonality

$$s(t) = \sum_{n=1}^{N} \left(a_n \cos\left(\frac{2\pi nt}{P}\right) + b_n \sin\left(\frac{2\pi nt}{P}\right) \right)$$

- h(t): Holiday

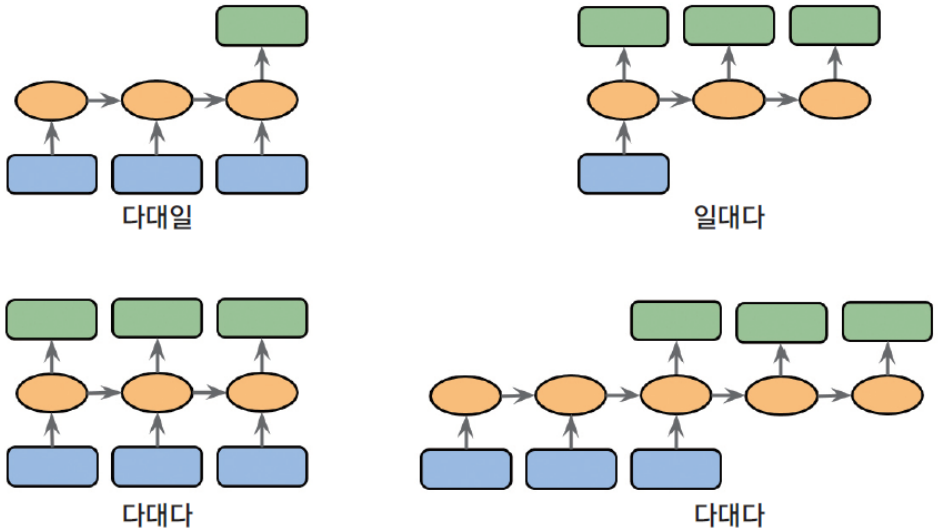
$$y(t) = g(t) + s(t) + h(t) + e_t$$

우리가 배운 걸로?

- SVM
- RF
- Boosting
- 등등등

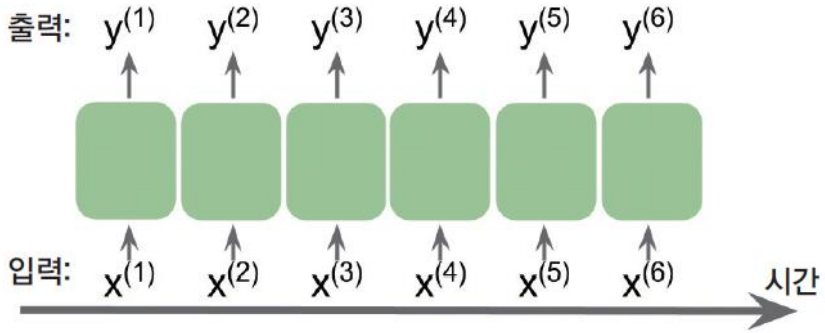
시퀀스 데이터

- 순서가 있는 데이터
  - ✓ 시계열은 시퀀스의 순서가 시간 기준인 특수한 경우
- 모든 시퀀스가 시간을 기준으로 나열되지는 않음
  - ✓ Ex. 텍스트, DNA 염기서열 등
- Input/output 의 형식에 따라 다양한 모델 설정

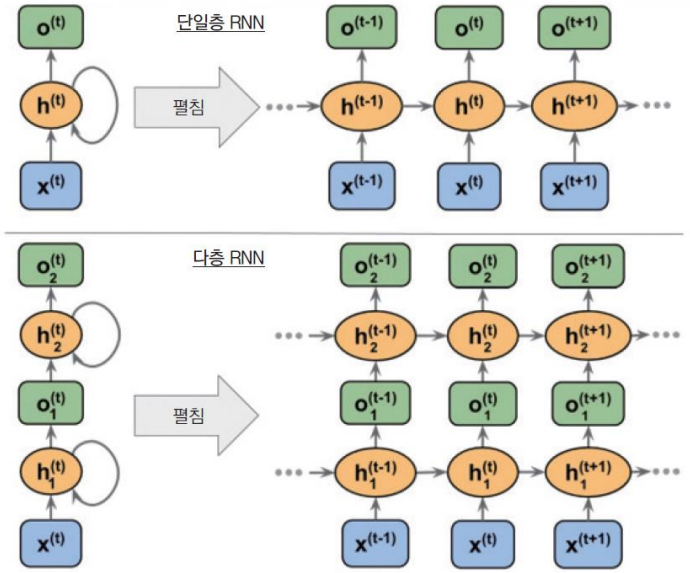


RNN의 기초

- 대전제
  - ✓ 이전의 상태가 지금의 상태(결과)에 영향을 끼친다
  - ✓ 미래의 상태는 이전 상태들의 영향을 합친 것이다



- 그렇다면, 이전 상태를 다음 학습에 반영해볼까?

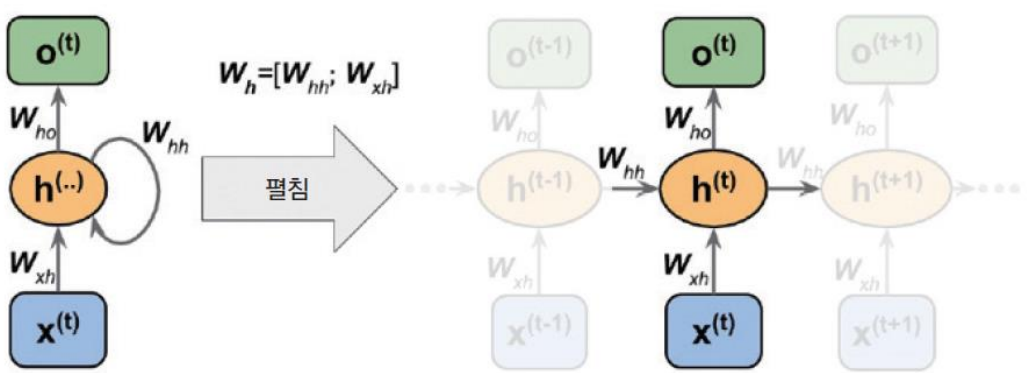


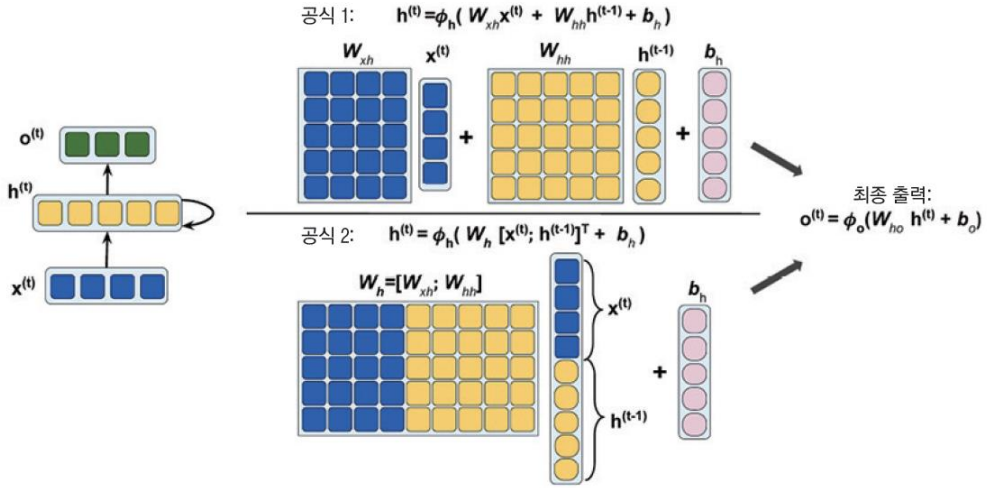
수식으로는

$$z_h^{(t)} = W_{xh}x^{(t)} + W_{hh}h^{(t-1)} + b_h$$

$$h^{(t)} = \phi_h\left(z_h^{(t)}\right) = \phi_h\left(W_{xh}x^{(t)} + W_{hh}h^{(t-1)} + b_h\right)$$

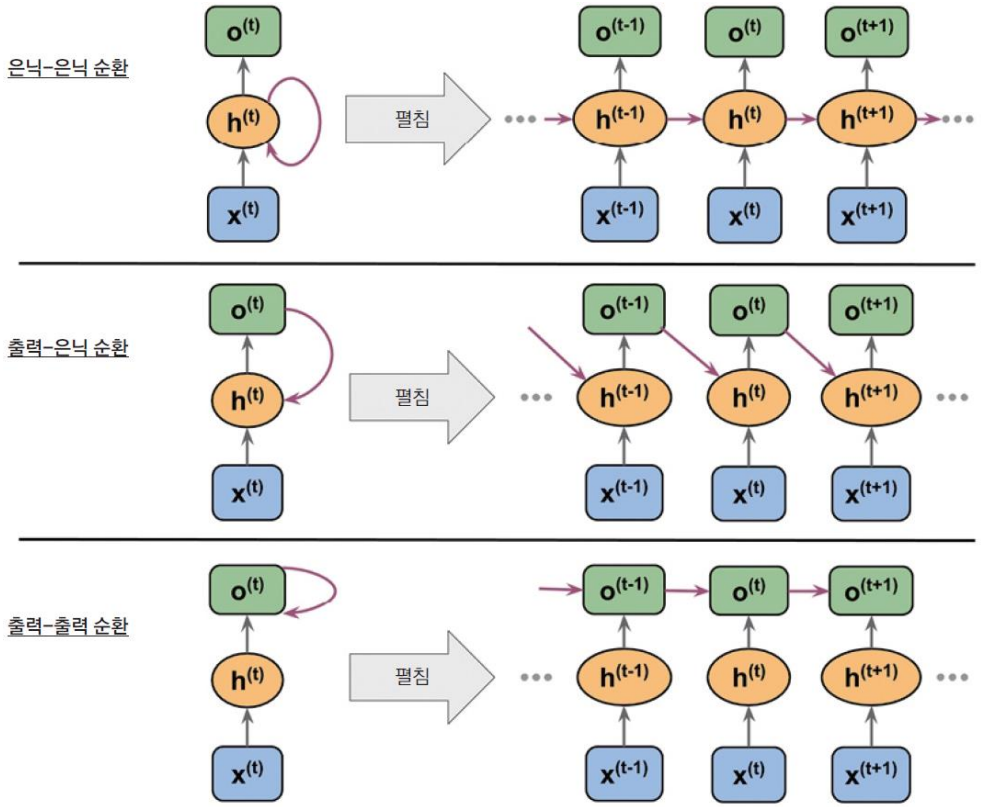
$$h^{(t)} = \phi_h\left([W_{xh}; W_{hh}]\begin{bmatrix} x^{(t)} \\ h^{(t-1)} \end{bmatrix} + b_h\right)$$





출력값도 순환에 사용해보자

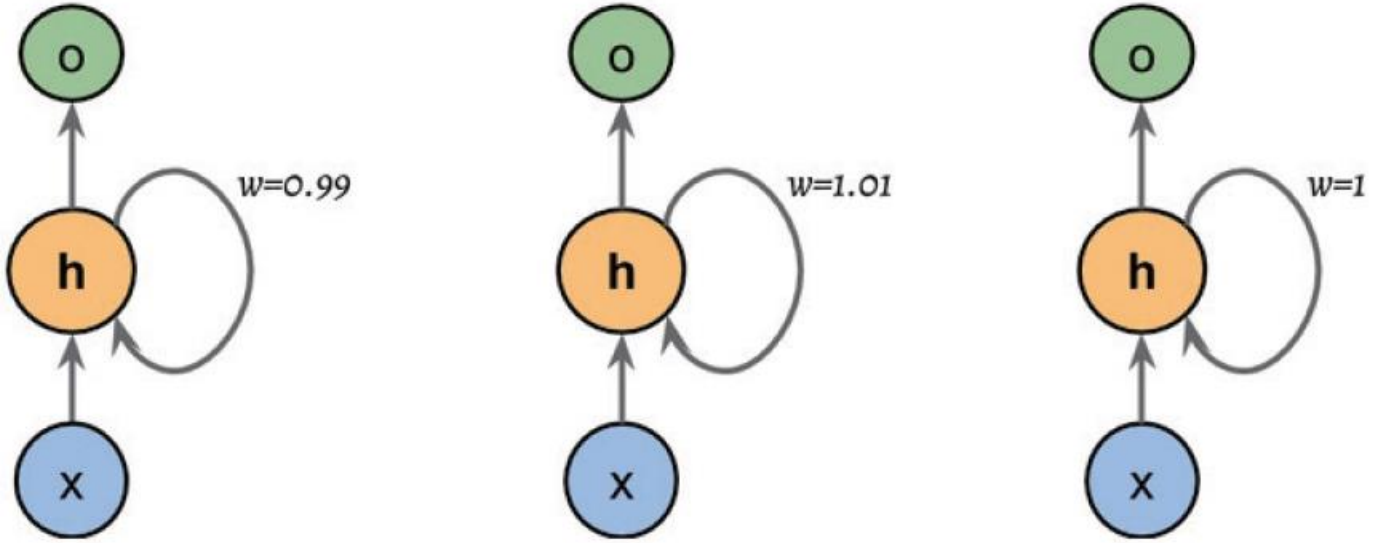
- 현재 타임 스텝에서 은닉층 ht에 추가(그림 16-7의 출력-은닉 순환)
- 현재 타임 스텝에서 출력층 ot에 추가(그림 16-7의 출력-출력 순환)



Gradient 가 사라진다

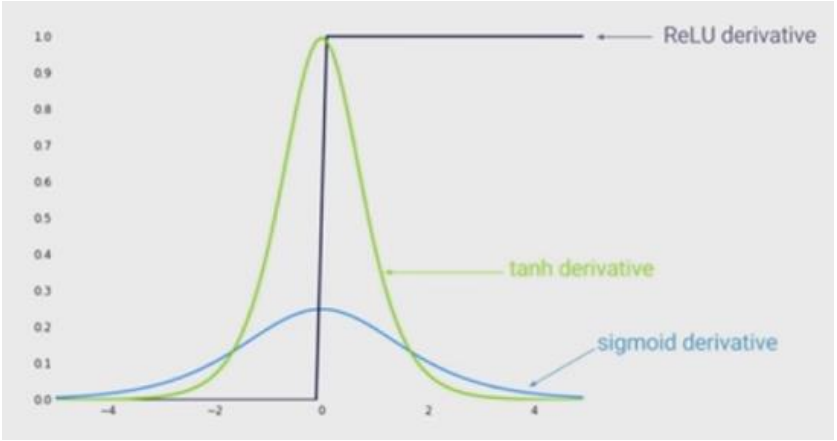
- Loop 의 weight 에 따라 h 에 미치는 영향이 결정됨
- 시퀀스가 길어질수록 w 에 민감하게 반응

그레이디언트 소실: $|w_{hh}| < 1$ 그레이디언트 폭주: $|w_{hh}| > 1$ 그레이디언트 유지: $|w_{hh}| = 1$



몇 가지 방법들

- Exploding Gradient (|W| > 1)
  - ✓ Gradient Clipping : 특정 값을 넘어가지 않도록 gradient 를 제한
- Vanishing Gradient (|W| < 1)
  - ✓ Better Activation Functions (ex. ReLU)
  - ✓ TBPTT (Truncated BackPropagation Through Time): 역전파의 타임 스텝 제한
  - ✓ Better Weight Initialization (with Identity Matrix)
  - ✓ LSTM (요건 조금 이따)
  - ✓ Architecture(skip connection, dense connection…)



https://i.stack.imgur.com/gMpB4.png

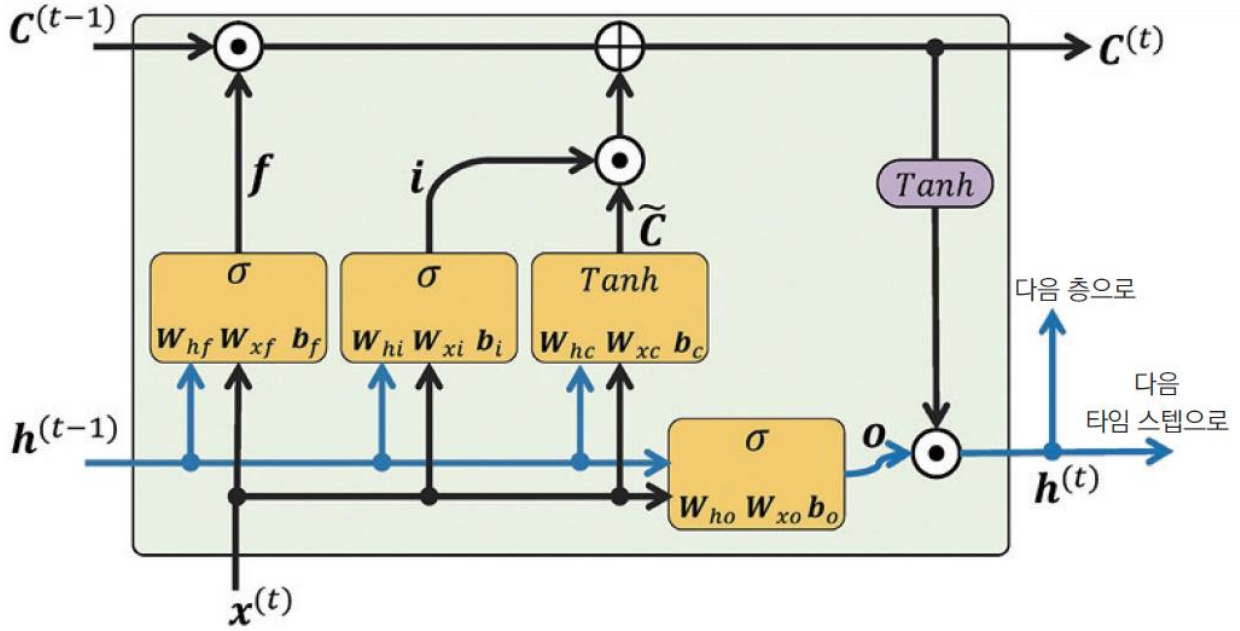
Long Short Term Memory

- RNN의 문제 -> Gradient 가 이상해짐
- 어떻게 이전 셀들의 효과를 유지할 것인가?
- 우리는 어떻게 기억을 유지하는가 생각해봅시다

LSTM의 구조

- C(t-1) : t-1의 셀 상태
- x(t): 타임 스텝 t에서 입력 데이터
- h(t-1): 타임 스텝 t -1에서 은닉 유닛의 출력
- ⊙ : 원소별 곱셈
- ⊕ : 원소별 덧셈

- Activation Function을 잘 보세요!
  - ✓ Sigmoid & tanh



LSTM의 Gates

- 삭제 게이트(ft)
  - ✓ 메모리 셀이 무한정 성장하지 않도록 셀 상태를 다시 설정
  - ✓ 삭제 게이트가 통과할 정보와 억제할 정보를 결정

$$\boldsymbol{f}_t = \sigma\left(\boldsymbol{W}_{xf}\boldsymbol{x}^{(t)} + \boldsymbol{W}_{hf}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_f\right)$$

- 입력 게이트(it)
  - ✓ 셀 상태를 업데이트 하는 역할

$$\boldsymbol{i}_t = \sigma\left(\boldsymbol{W}_{xi}\boldsymbol{x}^{(t)} + \boldsymbol{W}_{hi}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_i\right)$$

$$\tilde{\boldsymbol{C}}_t = \tanh\left(\boldsymbol{W}_{xc}\,\boldsymbol{x}^{(t)} + \boldsymbol{W}_{hc}\,\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_c\right)$$

$$\boldsymbol{C}^{(t)} = \left(\boldsymbol{C}^{(t-1)} \odot \boldsymbol{f}_t\right) \oplus \left(\boldsymbol{i}_t \odot \tilde{\boldsymbol{C}}_t\right)$$

LSTM의 Gates

- 출력 게이트(ot)
  - ✓ 은닉 유닛의 출력 값을 업데이트

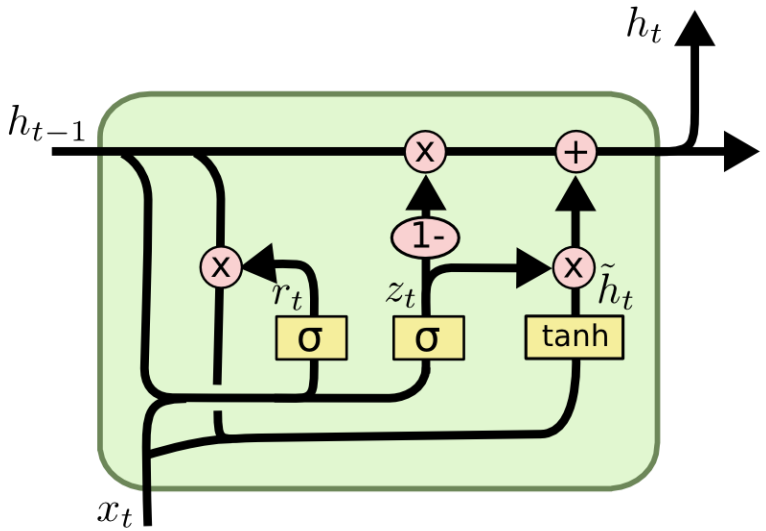
$$\boldsymbol{o}_t = \sigma\left(\boldsymbol{W}_{xo}\,\boldsymbol{x}^{(t)} + \boldsymbol{W}_{ho}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_o\right)$$

$$\boldsymbol{h}^{(t)} = \boldsymbol{o}_t \odot \tanh\left(\boldsymbol{C}^{(t)}\right)$$

- 실제 사용 시에 각각을 구현하지 않고 wrapper 함수 사용
  - ✓ tf.keras.layers.LSTM

GRU

- LSTM의 복잡도를 줄이고, cell state 정보를 hidden state 에 합체!
- 하지만 여전히 LSTM도 많이 사용됨



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

더 찾아보면 좋은 것들

- seq2seq
- Attention
- Transformers -> 특히 NLP, seq 생성 !