RNN

h(t-1) 대신 t-1 시점의 Weighted sum을 과거 정보로 사용 안 한 이유?

그냥 Weighted sum은 non-linear의 보장을 못 한다. backpropagation 시점에서 필요한 건 non-linear의 특성을 가진 activation function

그렇다고, hidden layer의 activation function의 의미가 중요한가?

hidden layer의 activation layer의 주 목적은 위에서도 말했듯이 non-linear의 특성 때문이다.
output 단계에서나 확률이나, 양수 값의 특정 목적을 가지고 사용을 하지 hidden layer에서는 의미가 크게 없다.

T 시점의 node가 가지는 의미

[T-1 시점의 activation 값 * weight(hh)] + [T 시점의 weighted sum * weight(xh)] 으로 구성
⇒ T-1 시점의 활성화 값과 현재 input feature 관계를 고려해 현재 노드를 구성한다.
activation 값은 의미 x →(T-1 시점의 weighted sum으로 고려)
⇒ T-1 시점의 weighted sum과 현재 weighted sum의 중요도(관계도=weight)를 고려해서 더해준다.
⇒ 즉, 노드는 T-1 시점의 이전 layer의 전체 노드와, T 시점의 이전 layer의 전체 노드들에게 얼마 만큼의 영향을 받고 구성되는가!

현재 layer의 노드가 여러개라면

그 layer의 노드들은 서로 독립적이다. why? 노드 = feature, 이 feature는 어떻게 구성되는가 → 이전 layer의 weighted sum으로 인해 구성된다. 각 노드가 가지고 있는 weight는 서로 다르게 구성 → 각 feature는 loss에 적합하게 각자의 weight가 조절되면서 구성된다.

RNN의 단점 : Gradient Vanishing

먼 시점의 output을 loss에 의존해 gradient descent를 적용시켰을 때, Vanishing 문제로 먼 과거 시점의 weight가 update가 되지 않는다. 즉, 서로 거리가 먼 시점이라면 서로 영향을 주지 않는다.

이게 문제가 되는가? 먼 과거의 정보가 항상 필수적인가?
먼 과거의 정보가 먼 미래의 output에게 영향을 주지 않는 data라고 가정했을 때,
이 vanishing gradient는 문제가 되지 않는다. 하지만, 현실에서의 data를 input으로 넣었을 때, model의 구조 상 멀어보이지만, 사실은 멀지 않은 관계가 많다.

또한, weight의 값은 작고, 그 gradient값도 작기 때문에, 조금만 멀리 떨어져도 gradient vanishing problem이 발생할 가능성이 농후하다.

h(t-1)은 T-1 시점의 정보만 가지고 있는 것이 아니라 이전 과거의 정보를 모두 담고 있다.