

# 꼼꼼한 딥러닝 논문 리뷰와 코드 실습

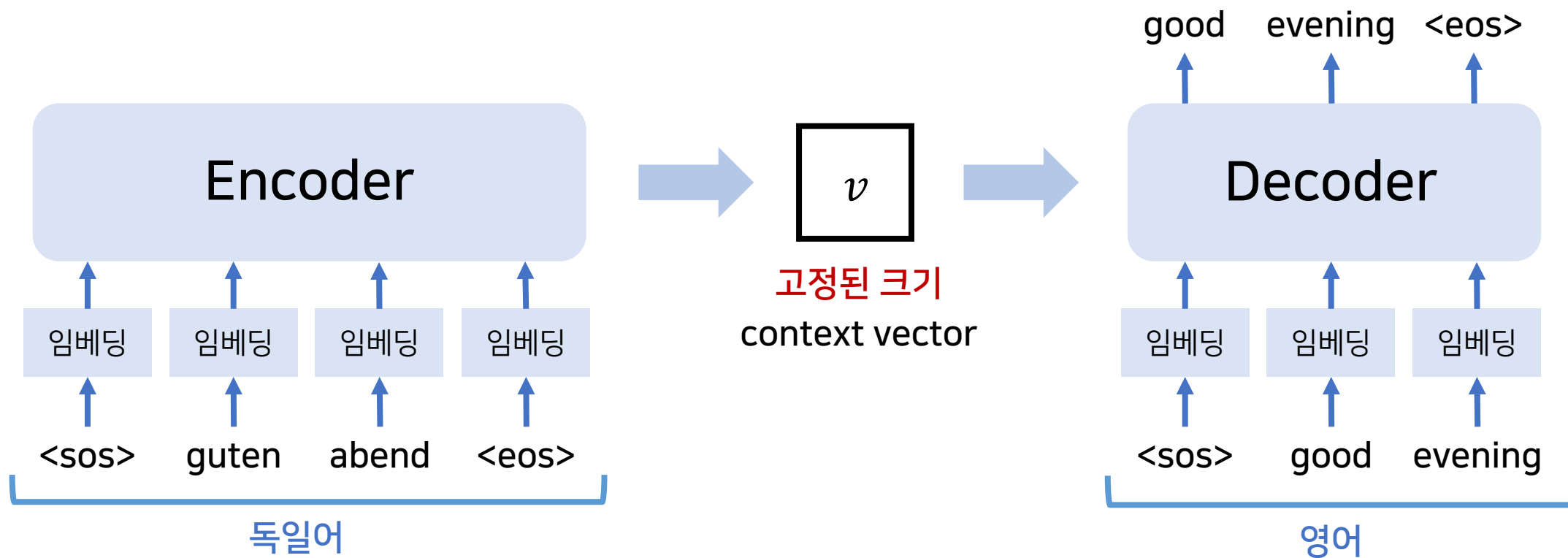
Deep Learning Paper Review and Code Practice

나동빈([dongbinna@postech.ac.kr](mailto:dongbinna@postech.ac.kr))

Pohang University of Science and Technology

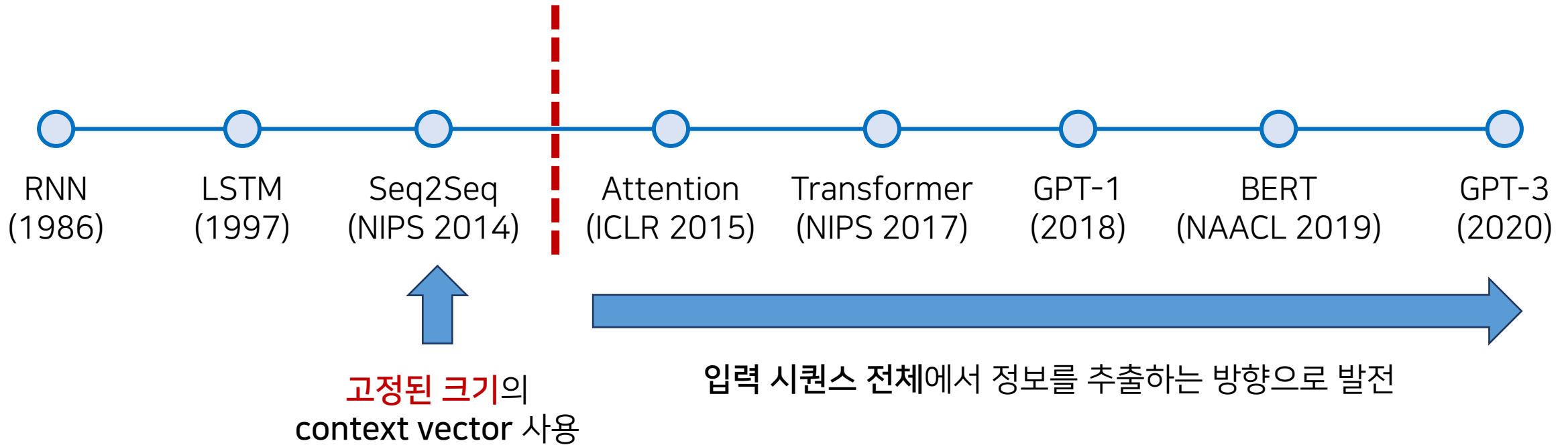
# Sequence to Sequence Learning with Neural Networks (NIPS 2014)

- 본 논문에서는 LSTM을 활용한 효율적인 Seq2Seq 기계 번역 아키텍처를 제안합니다.
  - Seq2Seq는 딥러닝 기반 기계 번역의 돌파구와 같은 역할을 수행했습니다.
  - Transformer(2017)가 나오기 전까지 state-of-the-art로 사용되었습니다.



## 딥러닝 기반의 기계 번역 발전 과정

- 2021년 기준으로 최신 고성능 모델들은 Transformer 아키텍처를 기반으로 하고 있습니다.
  - GPT: Transformer의 디코더(Decoder) 아키텍처를 활용
  - BERT: Transformer의 인코더(Encoder) 아키텍처를 활용



# 자연어 처리를 위한 기초 수학: 언어 모델(Language Model)

- 언어 모델이란 문장(시퀀스)에 확률을 부여하는 모델을 의미합니다.
- 언어 모델을 가지고 있으면 특정한 상황에서의 적절한 문장이나 단어를 예측할 수 있습니다.

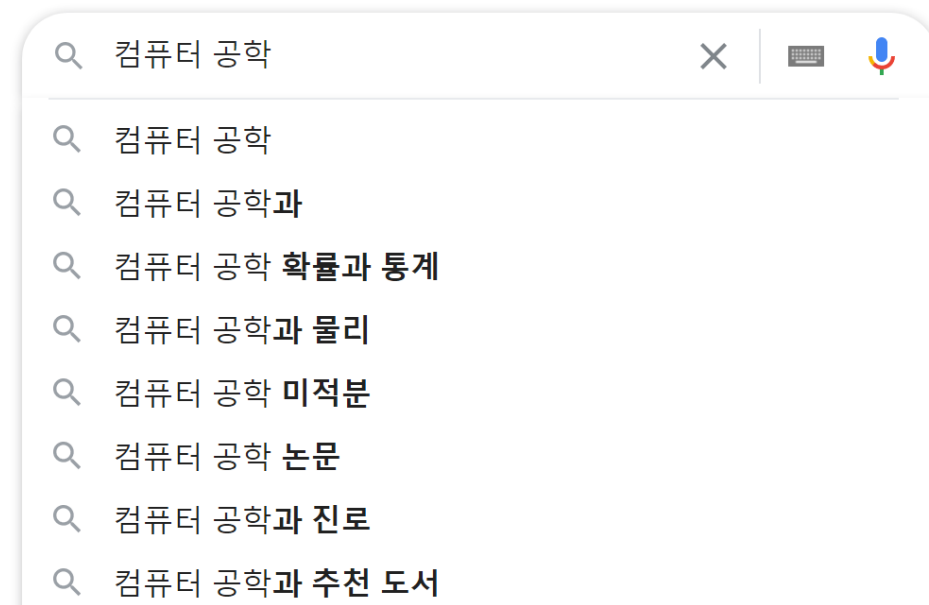
- 기계 번역 예시

- $P(\text{난 널 사랑해} | I \text{ love you}) > P(\text{난 널 싫어해} | I \text{ love you})$



- 다음 단어 예측 예시

- $P(\text{먹었다} | \text{나는 밥을}) > P(\text{싸웠다} | \text{나는 밥을})$



## 자연어 처리를 위한 기초 수학: 언어 모델(Language Model)

- 하나의 문장은( $W$ )은 여러 개의 단어( $w$ )로 구성됩니다.

- $P(W) = P(w_1, w_2, w_3, w_4, w_5, \dots, w_n)$

- $P(\text{친구와 친하게 지낸다}) = P(\text{친구와}, \text{친하게}, \text{지낸다})$

- 연쇄 법칙 (Chain Rule)

- $P(w_1, w_2, w_3, \dots, w_n) = P(w_1) * P(w_2|w_1) * P(w_3|w_1, w_2), \dots, P(w_n|w_1, w_2, \dots, w_{n-1})$

$$= \prod_{i=1}^n P(w_i|w_1, \dots, w_{i-1})$$

- $P(\text{친구와 친하게 지낸다}) = P(\text{친구와}) * P(\text{친하게}|\text{친구와}) * P(\text{지낸다}|\text{친구와 친하게})$

# 자연어 처리를 위한 기초 수학: 통계적 언어 모델(Statistical Language Model)

- 전통적인 통계적 언어 모델은 카운트 기반의 접근을 사용합니다.
  - $P(\text{지낸다}|\text{친구와 친하게}) = \frac{\text{count}(\text{친구와 친하게 지낸다})}{\text{count}(\text{친구와 친하게})}$
- 현실 세계에서 모든 문장에 대한 확률을 가지고 있으려면 매우 방대한 양의 데이터가 필요합니다.
- 예를 들어 “친구와 친하게”라는 시퀀스 자체가 학습 데이터에 존재하지 않으면 어떻게 될까요?
- 긴 문장은 처리하기가 매우 어렵습니다.
  - $P(\text{나는 공부를 마치고 집에서 밥을 먹었다}) = P(\text{나는}) * P(\text{공부를}|\text{나는}) * P(\text{마치고}|\text{나는 공부를}) * P(\text{집에서}|\text{나는 공부를 마치고}) * P(\text{밥을}|\text{나는 공부를 마치고 집에서}) * P(\text{먹었다}|\text{나는 공부를 마치고 집에서 밥을})$
- 현실적인 해결책으로 N-gram 언어 모델이 사용됩니다.
  - 인접한 일부 단어만 고려하는 아이디어입니다.

## 전통적인 RNN 기반의 번역 과정

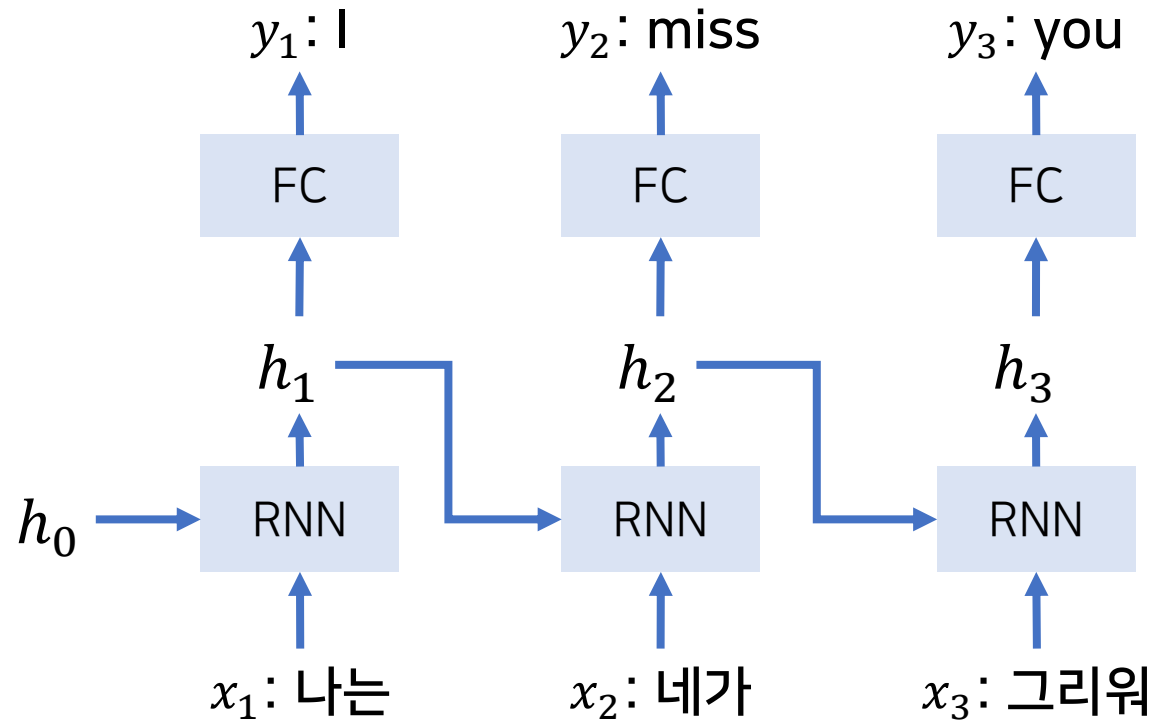
- 전통적인 초창기 RNN 기반의 언어 모델에서 번역이 이루어지는 과정은 다음과 같습니다.
- 전통적인 RNN 기반의 기계 번역은 입력과 출력의 크기가 같다고 가정합니다.

- 입력:  $(x_1, \dots, x_T)$

- 출력:  $(y_1, \dots, y_T)$

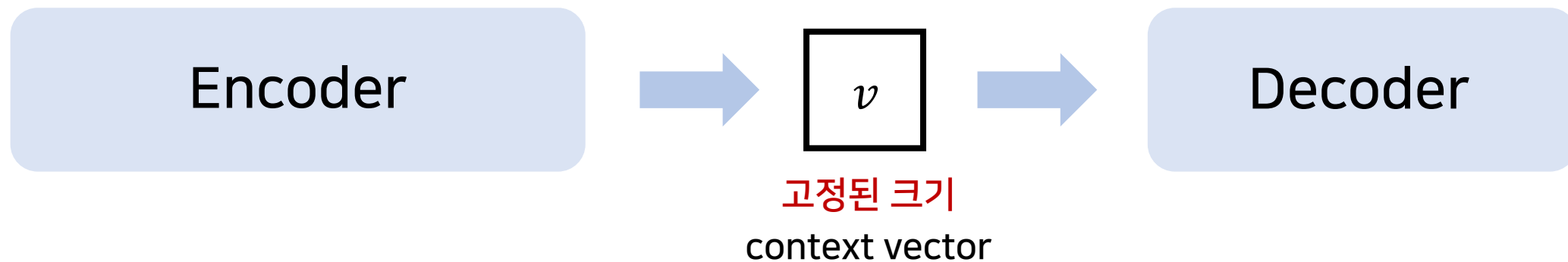
- $h_t = \text{sigmoid}(W^{hx}x_t + W^{hh}h_{t-1})$

- $y_t = W^{yh}h_t$



## RNN 기반의 Sequence to Sequence 개요

- 전통적인 초창기 RNN 기반의 언어 모델은 다양한 한계점을 가지고 있습니다.
  - 이를 해결하기 위해 인코더가 고정된 크기의 **문맥 벡터(context vector)**를 추출하도록 합니다.
  - 이후에 문맥 벡터로부터 디코더가 번역 결과를 추론합니다.
  - 본 Seq2Seq 논문에서는 LSTM를 이용해 문맥 벡터를 추출하도록 하여 성능을 향상시킵니다.
    - 인코더의 마지막 hidden state만을 context vector로 사용합니다.

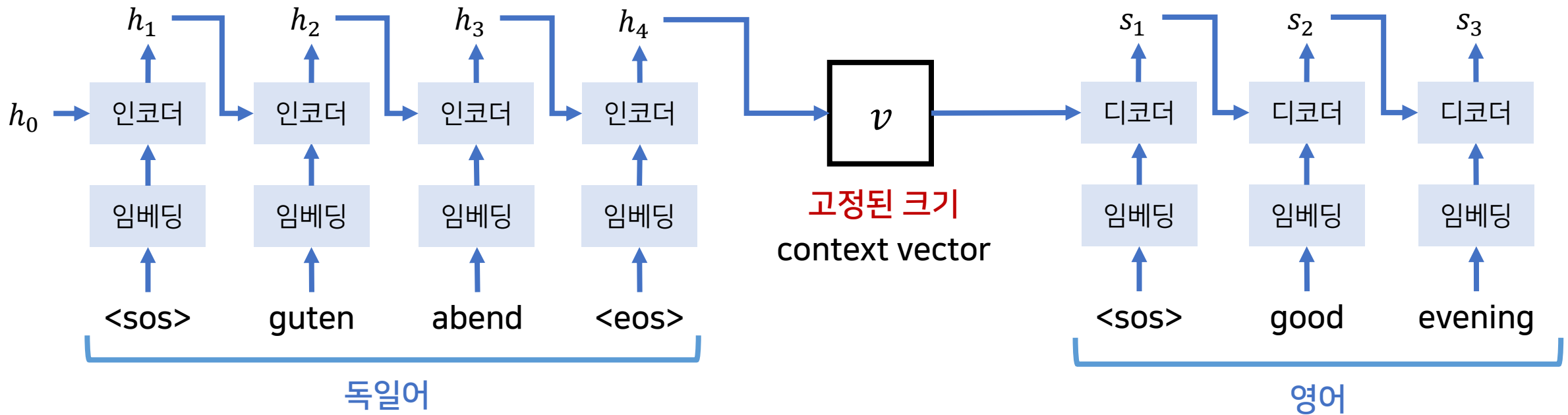


- 인코더와 디코더는 **서로 다른 파라미터(가중치)**를 가집니다.



# RNN 기반의 Sequence to Sequence 자세히 살펴보기

- $x_t$  : 현재의 입력 단어
- $h_t$  : 지금까지 입력된 문장에 대한 정보를 담은 벡터 표현
- $s_t$  : 지금까지 출력된 문장에 대한 정보를 담은 벡터 표현
- $y_t$  : 현재의 출력 단어

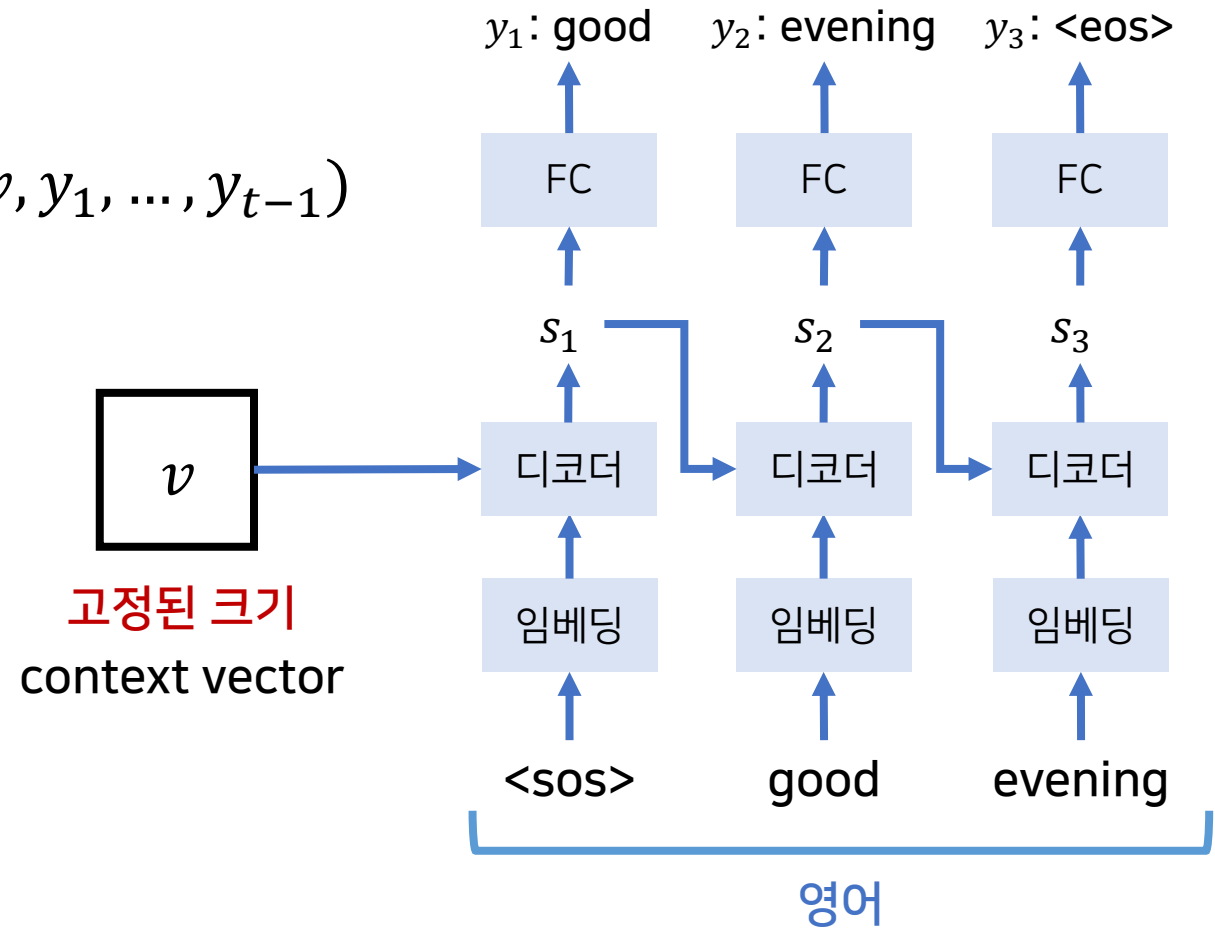


# RNN 기반의 Sequence to Sequence 자세히 살펴보기

- RNN 기반 Seq2Seq 모델의 목표 공식(formulation)은 다음과 같습니다.
  - 종료 시점:  $y_t = \langle \text{eos} \rangle$

$$p(y_1, \dots, y_{T'} | x_1, \dots, x_T) = \prod_{t=1}^{T'} p(y_t | v, y_1, \dots, y_{t-1})$$

- $v : x_1, \dots, x_T$ 에 대한 정보를 담은 벡터 표현
- $y_t$  : 현재의 출력 단어



## Seq2Seq의 성능 개선 포인트: LSTM 활용 및 입력 문장의 순서 뒤집기

- 기본적인 RNN 대신에 LSTM을 활용했을 때 더 높은 정확도를 보입니다.
- 실제 학습 및 테스트 과정에서 입력 문장의 순서를 거꾸로 했을 때 더 높은 정확도를 보입니다.
  - 출력 문장의 순서는 바꾸지 않습니다.

