Neural Machine Translation

Knowledge and Language Engineering Lab
이 원 기 (wklee@postech.ac.kr)

POSTECH
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

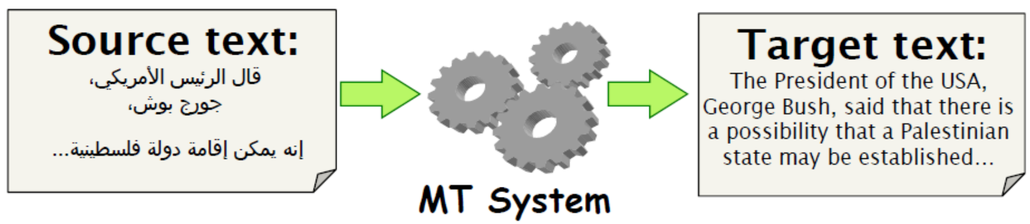
Goal

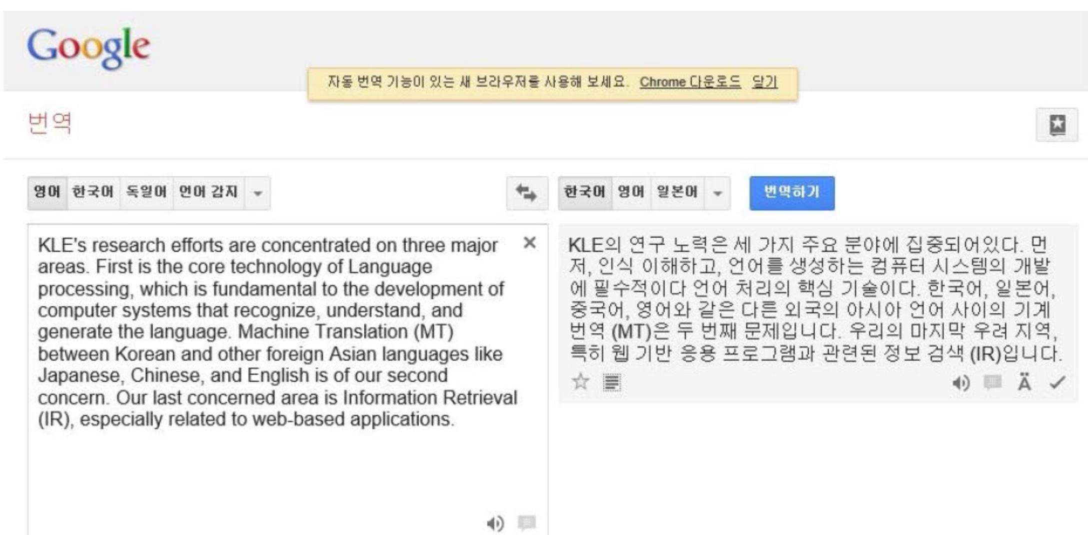
- 신경망 기계번역 모델에 대한 이해 및 구현
  - RNN 기반의 Sequence-to-Sequence 모델
  - Attention 메커니즘
- Torchtext
  - 효율적인 텍스트 데이터 (전) 처리를 위한 라이브러리

NEURAL MACHINE TRANSLATION

기계 번역

- 기계번역이란?
  - 원시문 (Source text) 를 **다른 언어**로 된 대상문장 (Target text) 로 **자동 번역**하는 기술

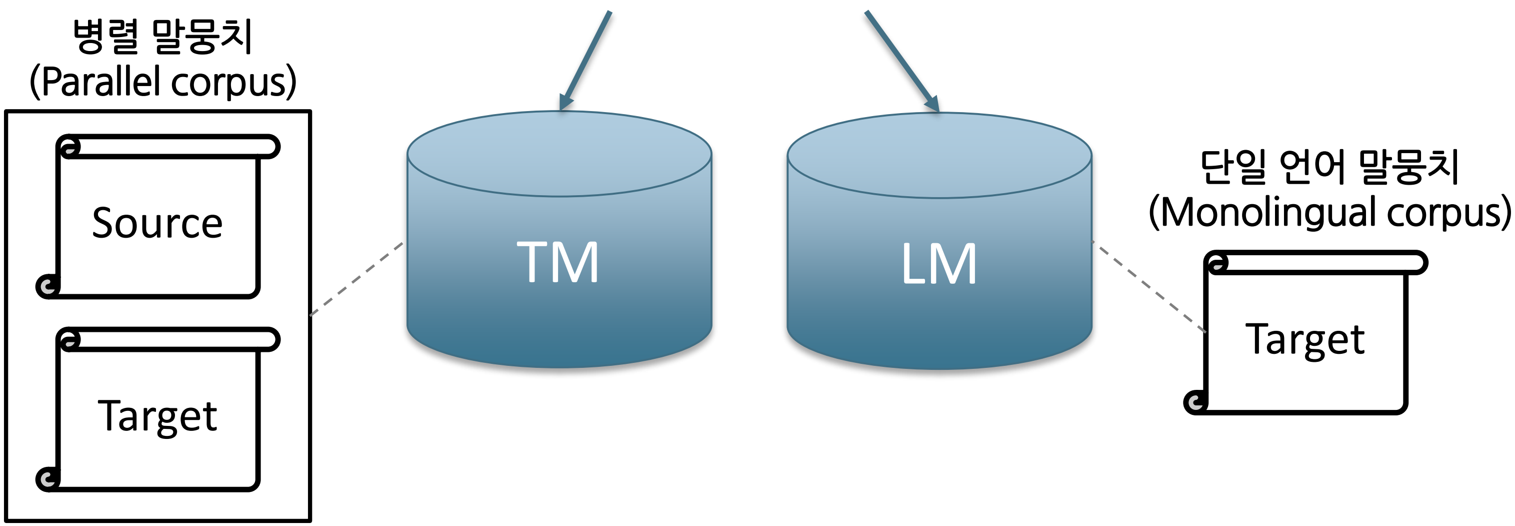




기존의 방법론

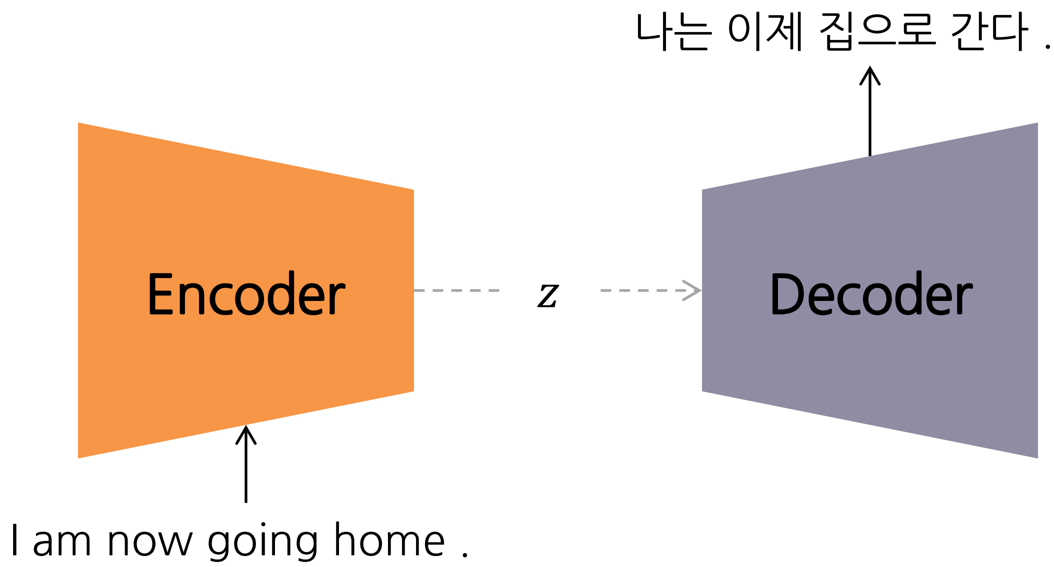
- 통계 기반 기계 번역 (Statistical Machine Translation: SMT)
  - 원시문과 대상문에 대한 확률식 정의
    - 원시문(ex. 영어) : $X = \{x_1, x_2, \dots, x_n\}$
    - 대상문(ex. 한국어): $Y = \{y_1, y_2, \dots, y_n\}$

$$\rightarrow P(Y|X) \cong P(X|Y) \cdot P(Y)$$



신경망 기반 기계번역

- Sequence-to-Sequence 모델
  - Encoder
    : 무작위 길이의 Source 문장을 고정된 길이의 Vector로 변환
  - Decoder
    : Encoder 의 Vector 정보와 입력(Input) 를 이용하여
    가변 길이의 문장 생성



신경망 기반 기계번역

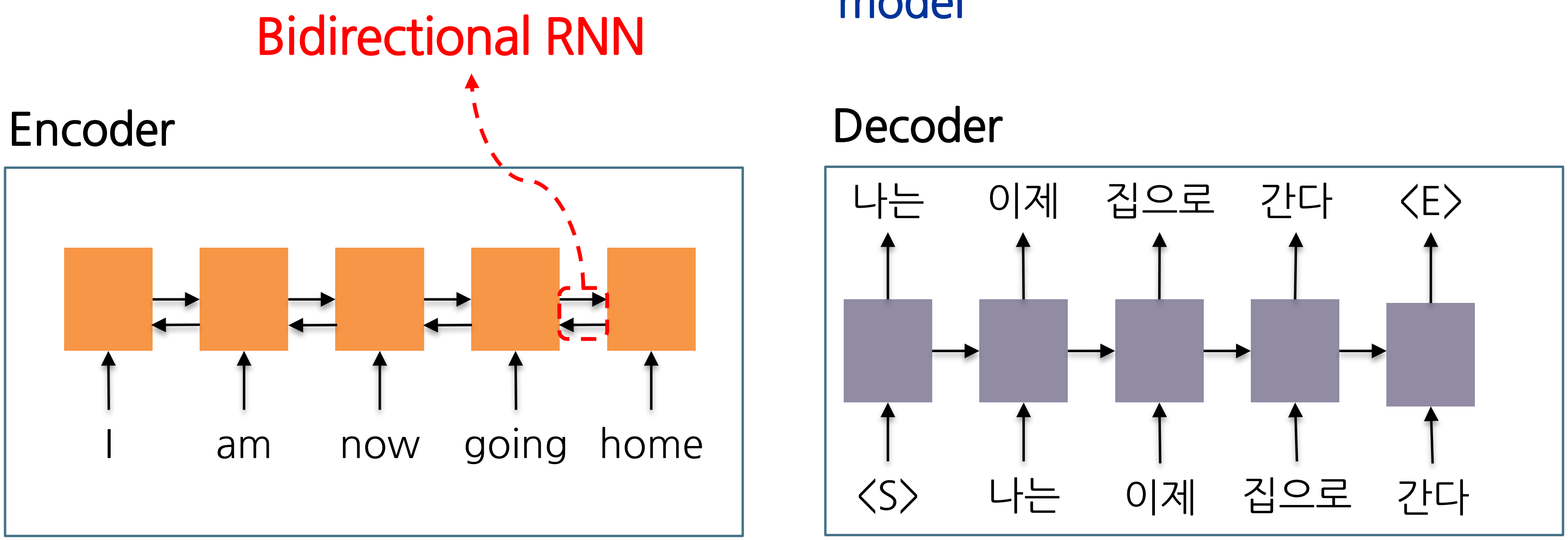
- 문제 정의
  - $y_{1:n}^{*} = argmax\ P(y_{1:n}|x_{1:m})$

    $= argmax\ P(y_1|x_{1:m}) \cdot P(y_2|y_1, x_{1:m}) \cdot P(y_3|y_2, y_1, x_{1:m}) \cdots P(y_n|y_{1:n-1}, x_{1:m})$

    $= argmax\ \prod_{t=1}^{n} P(y_t|\{y_1, \ldots, y_{t-1}\}, x_{1:m})$

신경망 기반 기계번역

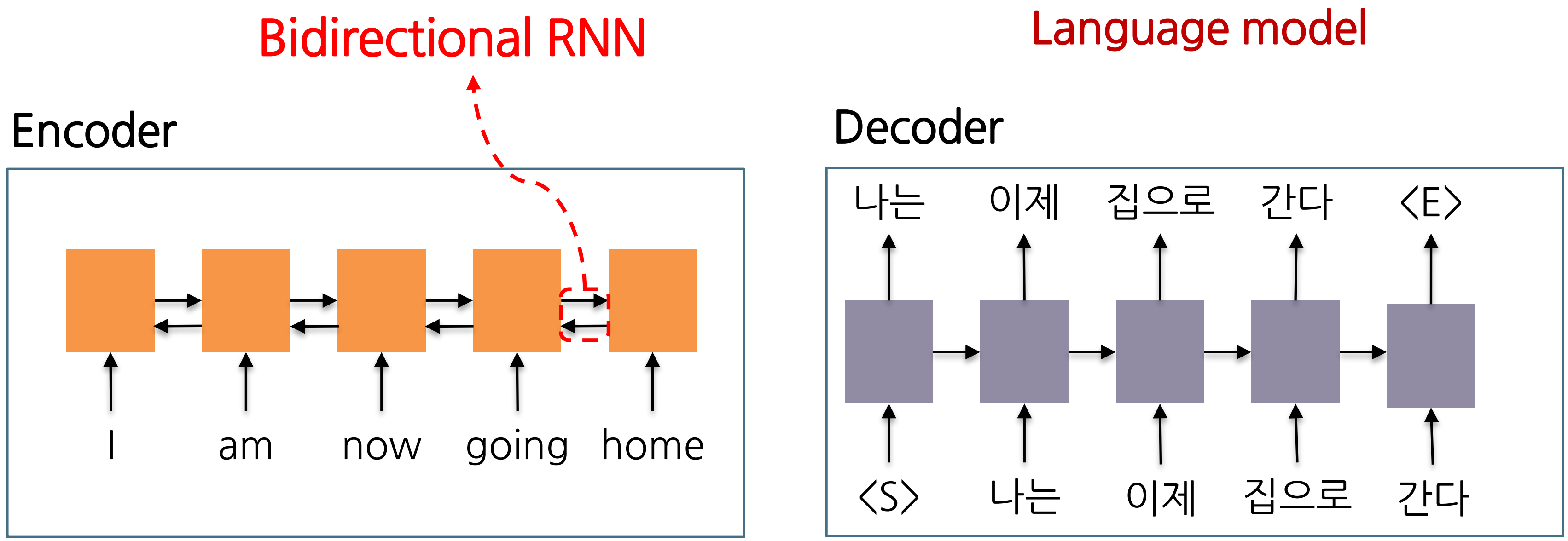
- RNN 기반의 Sequence-to-Sequence 모델
  - $y^*_{1:n} = argmax \prod_{t=1}^{n} P(\underbrace{y_t|\{y_1,\dots,y_{t-1}\}}_{\text{Language model}}, \underbrace{x_{1:m}}_{\text{Encoder}})$



신경망 기반 기계번역

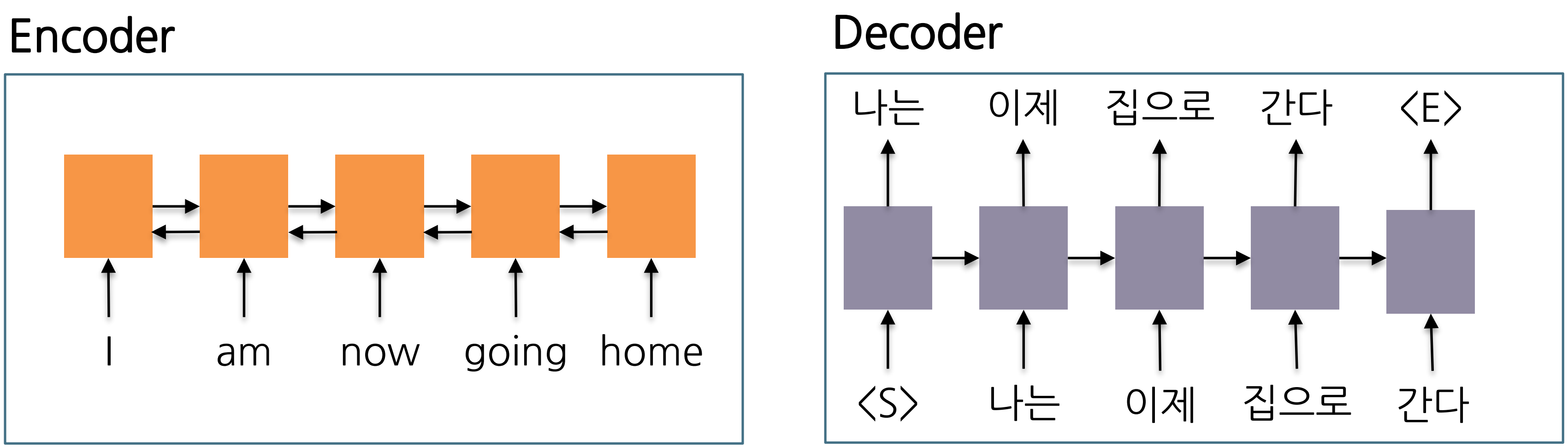
- RNN 기반의 Sequence-to-Sequence 모델
  - $y^*_{1:n} = argmax \prod_{t=1}^{n} \boxed{P(y_t|\{y_1, \dots, y_{t-1}\}, x_{1:m})}$

Conditional
Language model



신경망 기반 기계번역

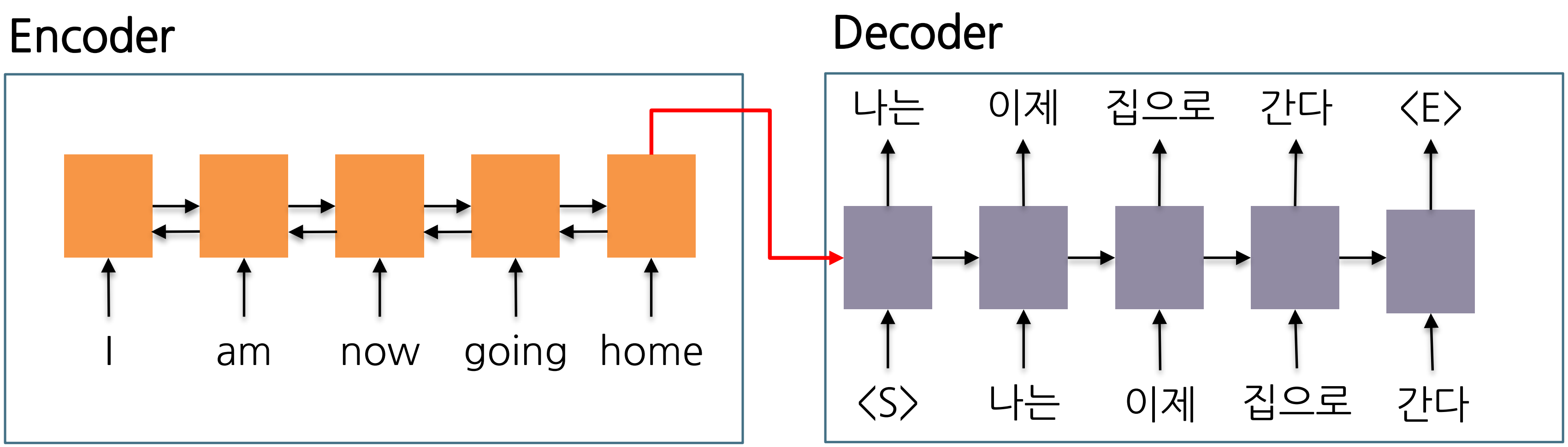
- RNN 기반의 Sequence-to-Sequence 모델
  - Encoder로 부터 입력열에 대한 **고정 길이 벡터** 생성 방법 ?



신경망 기반 기계번역

- RNN 기반의 Sequence-to-Sequence 모델
  - Encoder로 부터 입력열에 대한 **고정 길이 벡터** 생성 방법 ?
  - 가장 쉬운 방법
    - **Encoder의 마지막 Hidden**으로 Decoder의 Hidden을 **초기화**

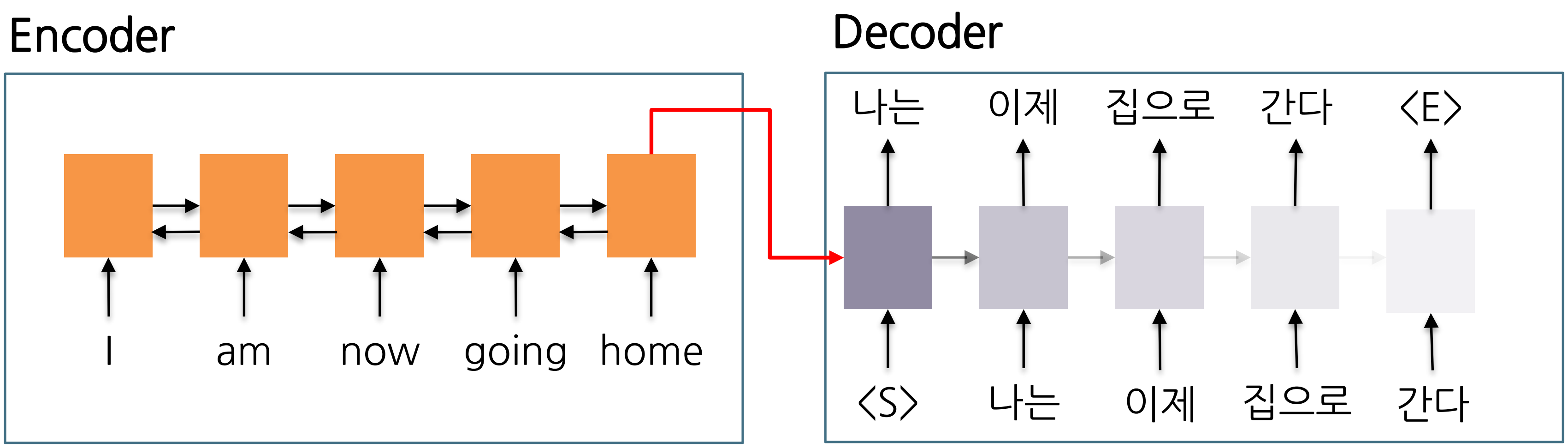
$\rightarrow h_t^{Enc} = f(x_1, x_2, .., x_n)$



신경망 기반 기계번역

- 장기 의존성 학습 어려움: Long-term dependency
  - Decoder의 **time-step이 증가할 수록** Encoder의 정보 전달이 약해짐.

    ➔ Encoder의 정보 활용이 부족.



신경망 기계번역

- Attention
  - Concept
    y_t를 번역(생성) 할 때 Encoder의 전체 입력열을 중요도에 따른 비율로 참조하는 것.
  - 중요도 (가중치) 에 따라 Encoder의 전체 입력열을
    압축된 Vector로 변환 (a.k.a. 'Context vector')

신경망 기계번역: Attention

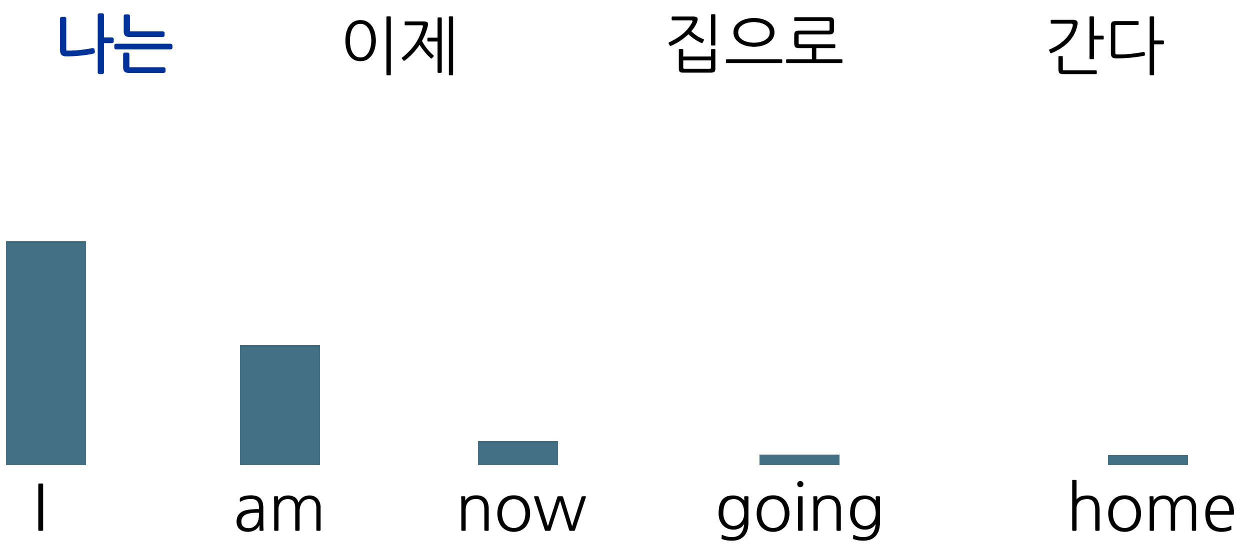
- Concept

나는 이제 집으로 간다

I am now going home

신경망 기계번역: Attention

- Concept

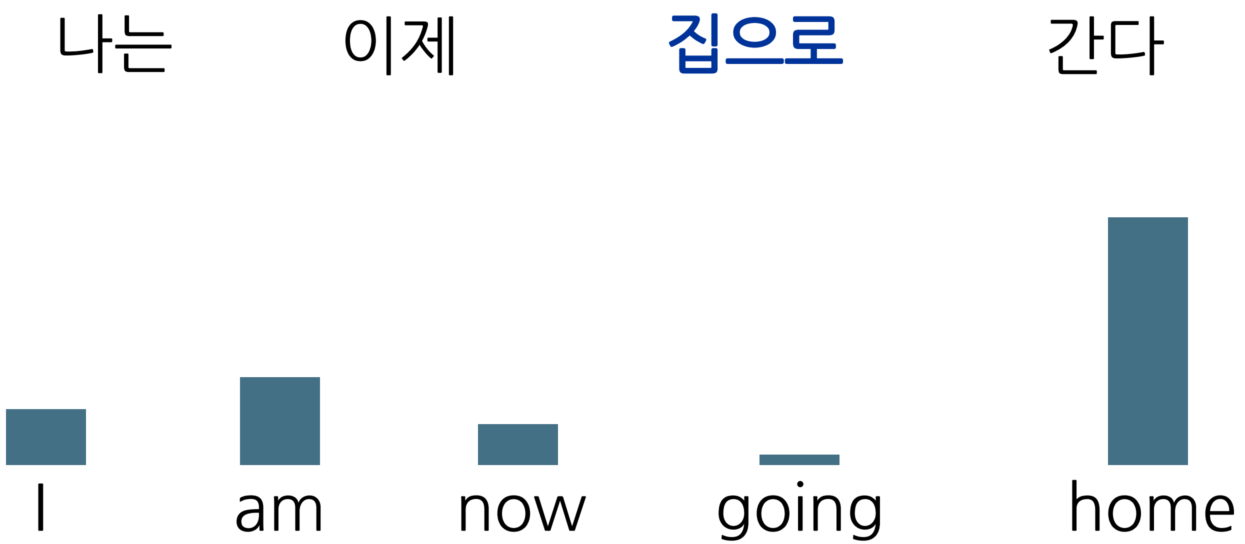


신경망 기계번역: Attention

- Concept

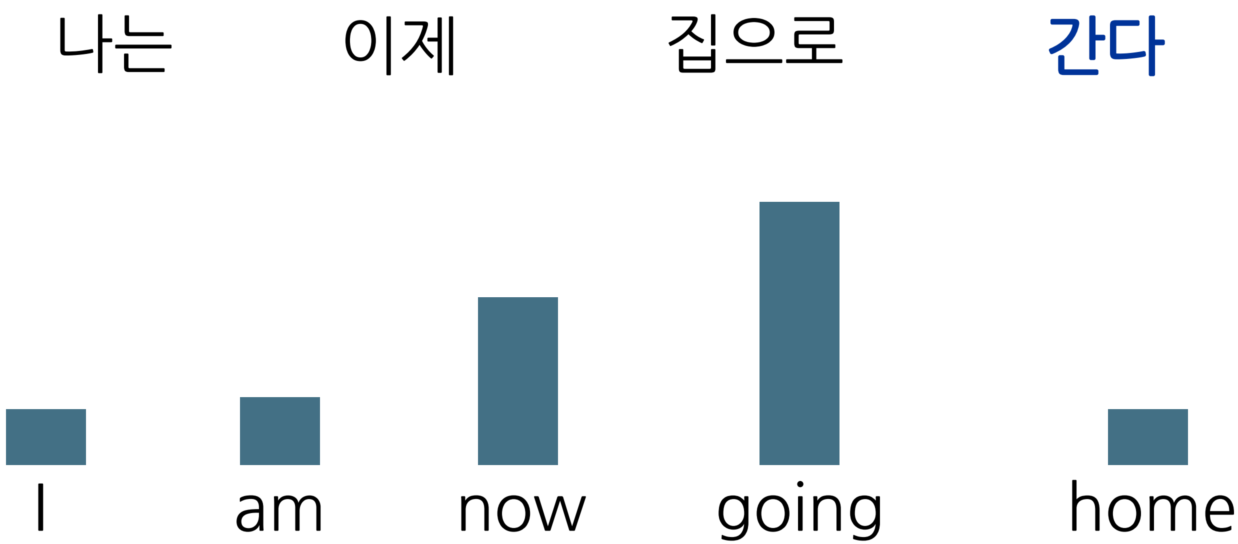
신경망 기계번역: Attention

- Concept



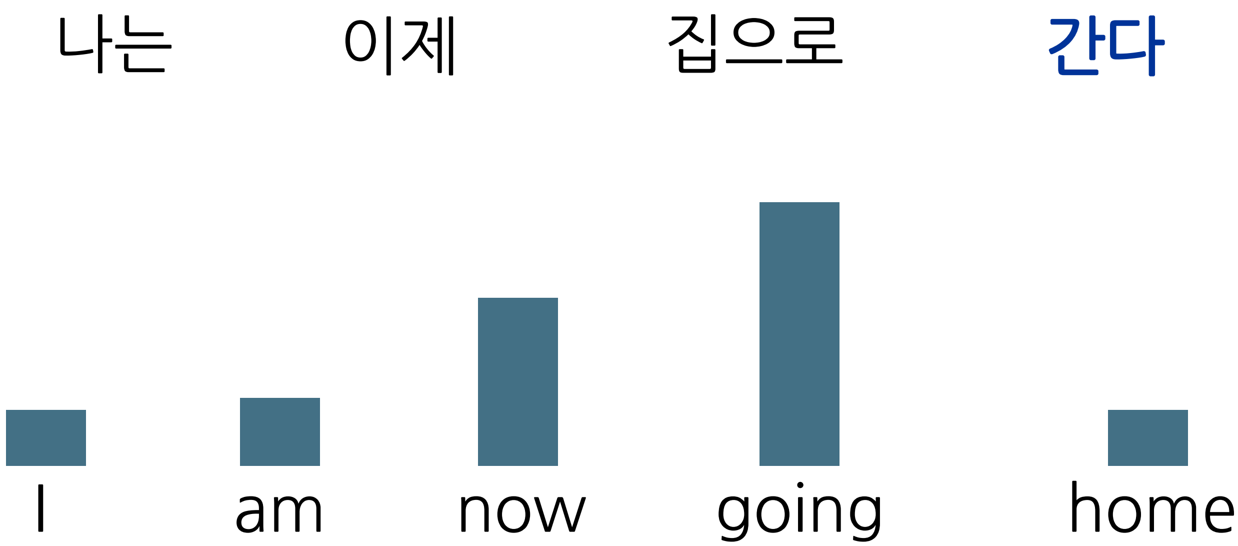
신경망 기계번역: Attention

- Concept



신경망 기계번역: Attention

- Concept



$$0.05 \times \boxed{\boldsymbol{h_1}} + 0.1 \times \boxed{\boldsymbol{h_2}} + 0.25 \times \boxed{\boldsymbol{h_3}} + 0.4 \times \boxed{\boldsymbol{h_4}} + 0.2 \times \boxed{\boldsymbol{h_5}} = \boxed{\mathbf{C_{vector}}}$$

신경망 기계번역: Attention

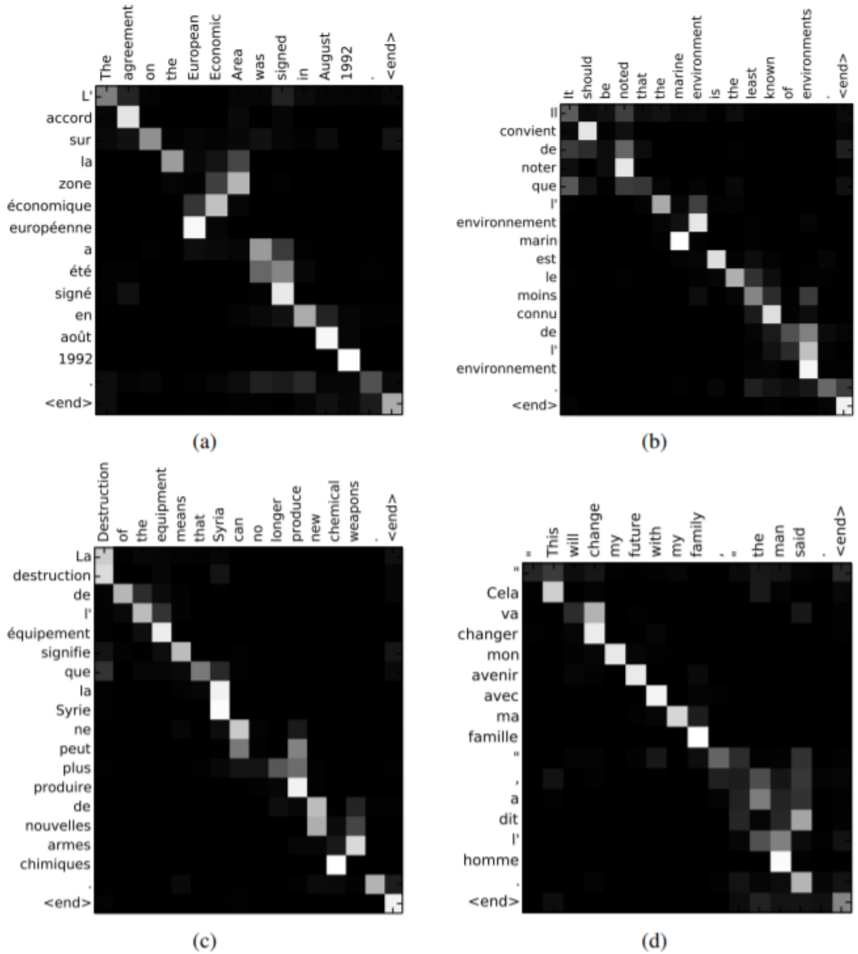


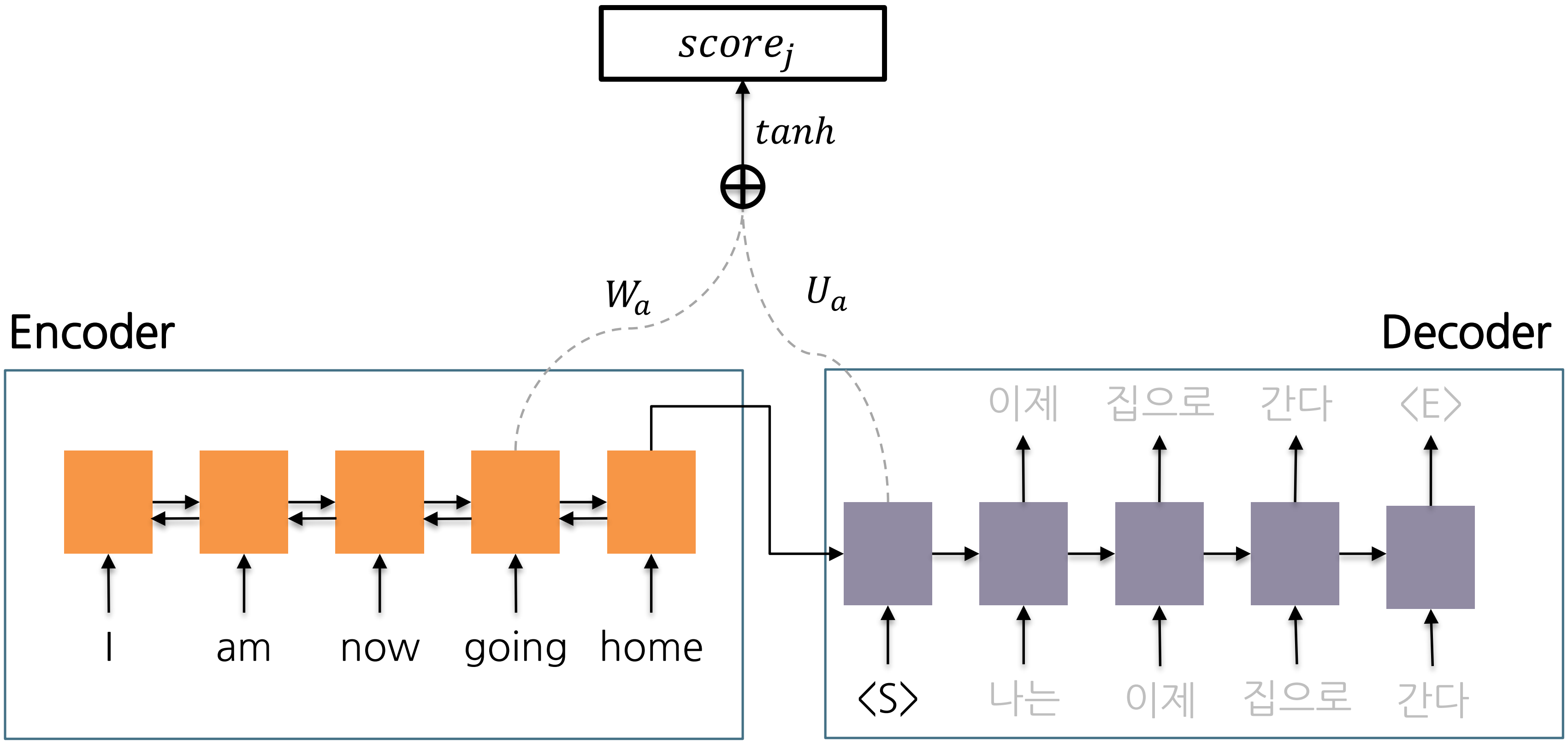
Figure 3: Four sample alignments found by RNNsearch-50. The x-axis and y-axis of each plot correspond to the words in the source sentence (English) and the generated translation (French), respectively. Each pixel shows the weight α_{ij} of the annotation of the j-th source word for the i-th target word (see Eq. (6)), in grayscale (0: black, 1: white). (a) an arbitrary sentence. (b–d) three randomly selected samples among the sentences without any unknown words and of length between 10 and 20 words from the test set.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio.

"Neural machine translation by jointly learning to align and translate."

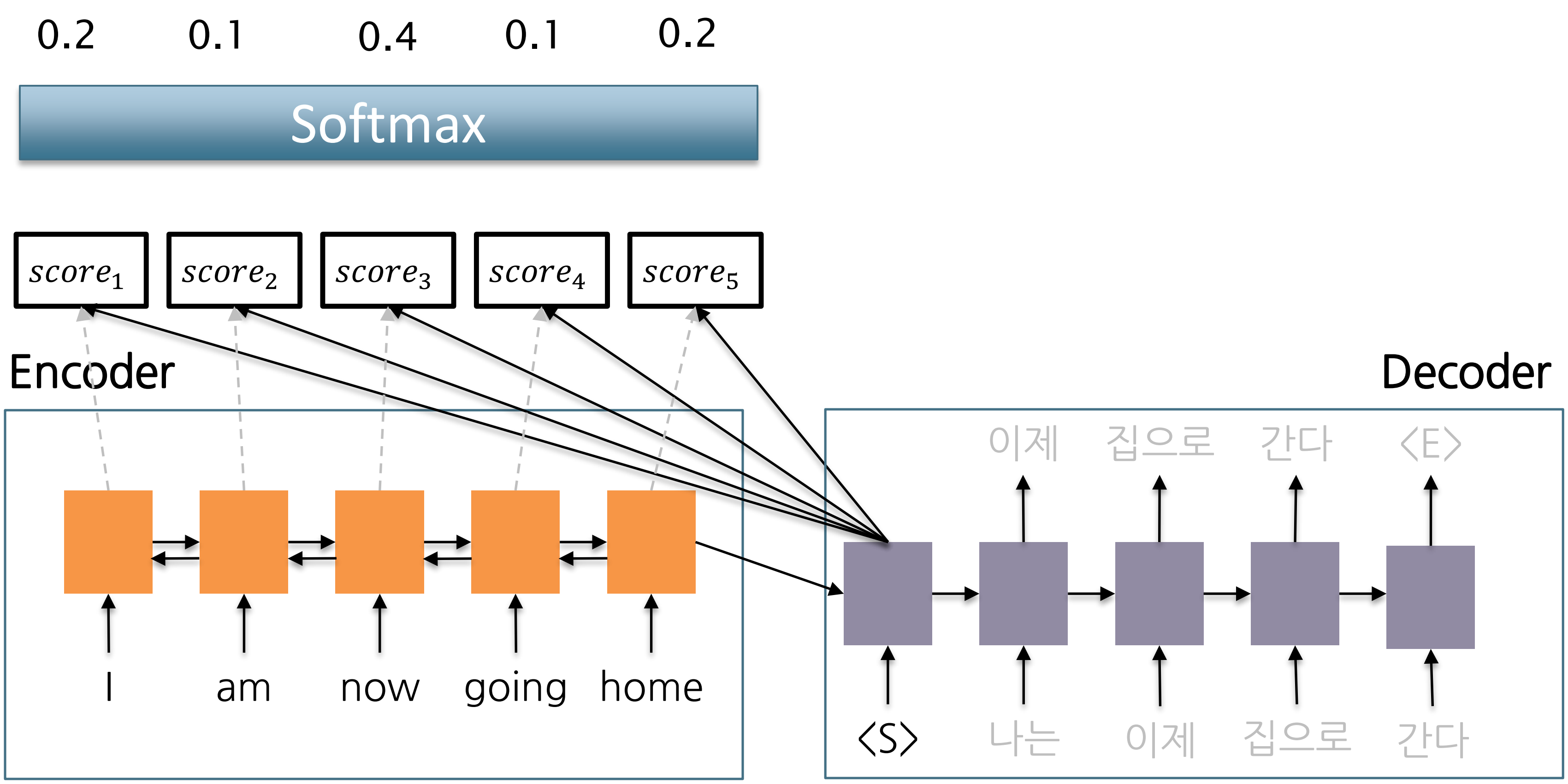
신경망 기계번역: Attention

- In practice: MLP approach
  - $\text{score}(h_i^{Enc}, h_t^{Dec}) = v_a^T tanh(W_a h_j^{Enc} + U_a h_t^{Dec})$ → scalar



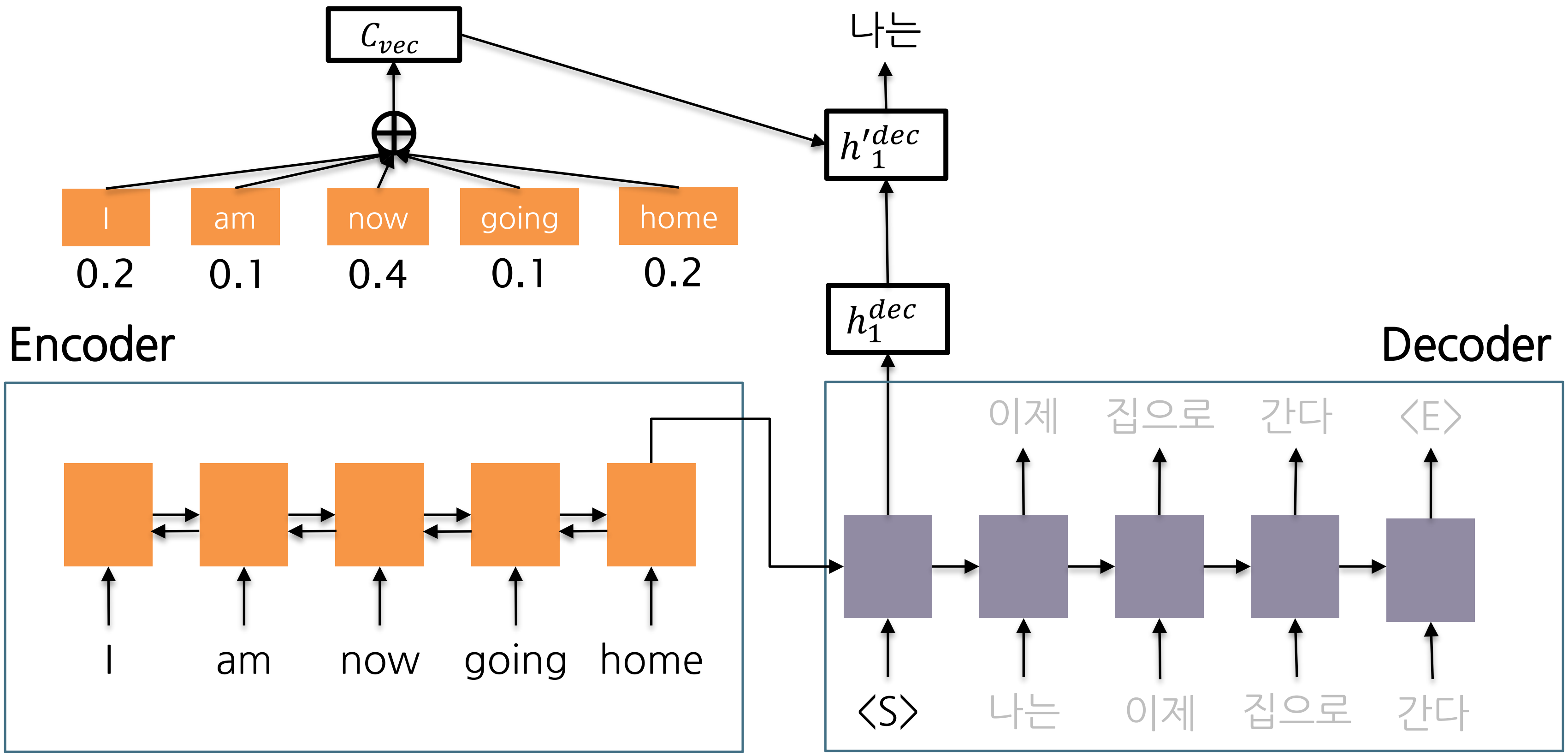
신경망 기계번역: Attention

- In practice: MLP approach
  - $\text{score}\left(h_j^{Enc}, h_t^{Dec}\right) = v_a^T tanh(W_a h_j^{Enc} + U_a h_t^{Dec})$



신경망 기계번역: Attention

- In practice: MLP approach
  - $C_{vec} = \sum_{j=1}^{|x|} a_j \cdot h_j^{Enc}$

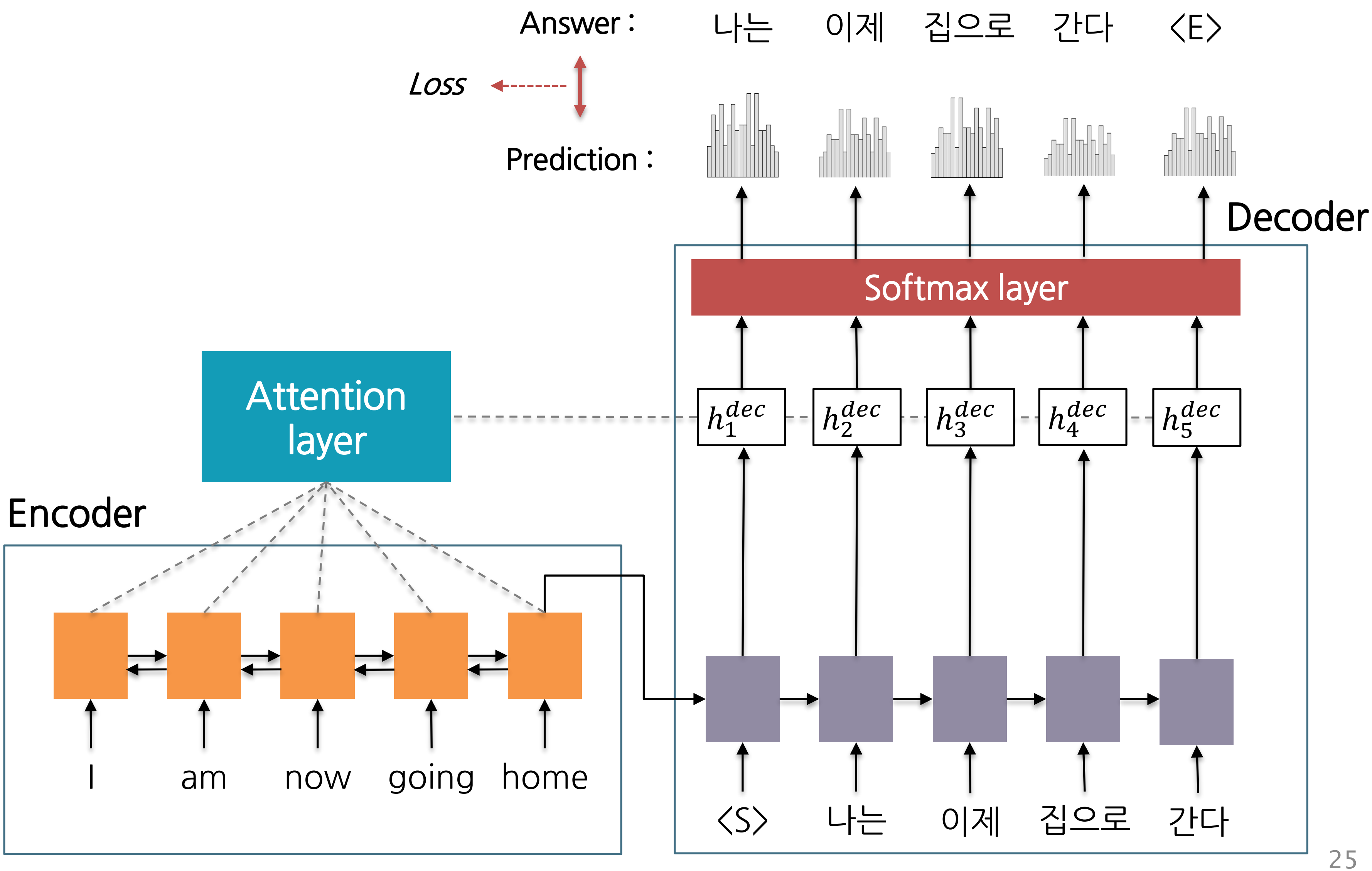


신경망 기계번역: Training

- Output probability:
  - $P(y_t|\{y_1, \ldots, y_{t-1}\}, x_{1:m}) \cong g(y_{t-1}, h_t^{dec}, c_t)$
    - where
      - y_{t-1} = Current input (t-1) for the decoder
      - h_t^{dec} = Hidden stage for time=t
      - c_t = Context vector for time=t
      - g = Nonlinear function representing the decoder

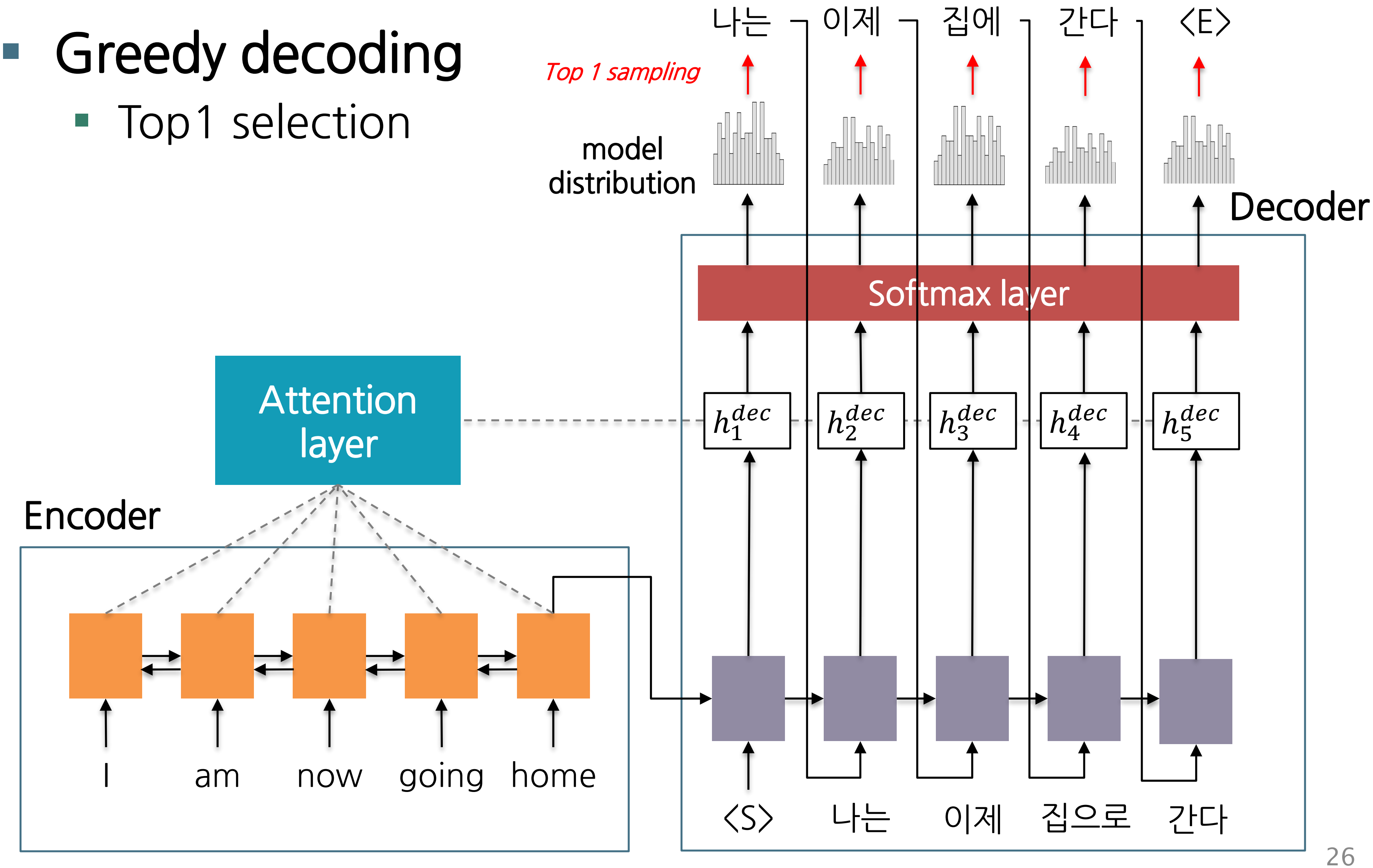
- Training objective:
  - $\mathcal{L}(\mathrm{y}) \;= -\frac{1}{T}\sum_{i=1}^{T} P(y_i) \log(P(\hat{y}_i))$
    - where
      - y_i = true distribution & $\hat{y}_i$ = model distribution

신경망 기계번역: Training



신경망 기계번역: Decoding (Inference)

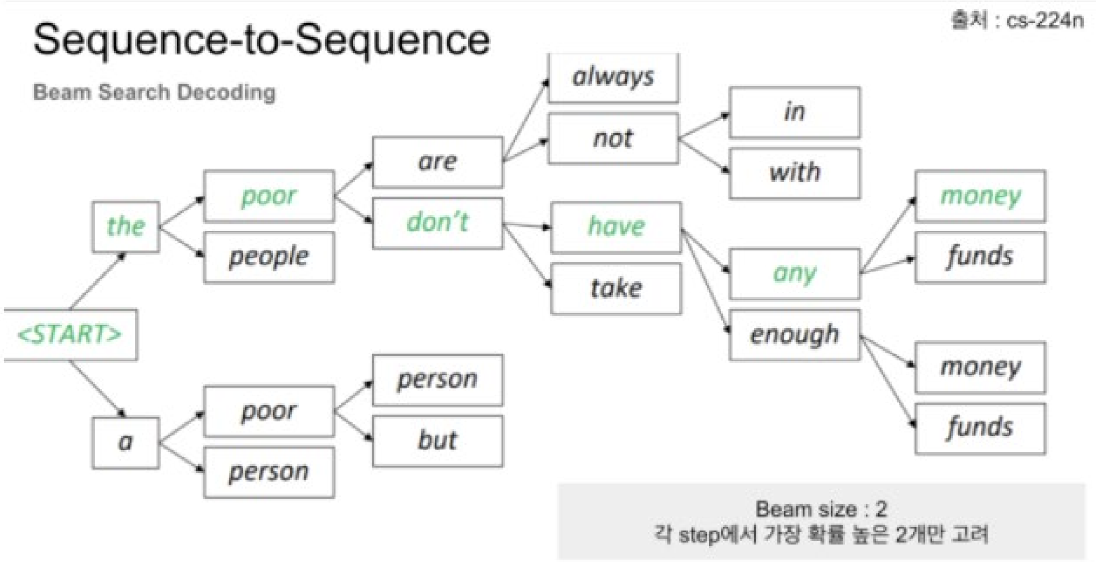
- **Greedy decoding**
  - Top1 selection



신경망 기계번역: Decoding (Inference)

- Beam search decoding
  - 각 time step 에서 가장 좋은 결과만을 선택 ≠ 최적의 문장열 (=Greedy decoding)
  - 각 time step 에서 모든 출력 시퀀스 검색 (Dynamic programming): Time complexity 로 인해 불가능

➔ Beam search: 탐색 영역을 각 time step 에서 top N (=beam size) 개로 제한하는 것.



PRACTICE:
NMT 모델 구현하기

Requirement

- pip install torchtext
- pip install mosestokenizer
- pip install torch

PRACTICE#2: OPEN SOURCE TOOLKIT

Useful tools

- Fairseq (Pytorch): https://github.com/pytorch/fairseq
- OpenNMT-py (Pytorch): https://github.com/OpenNMT/OpenNMT-py
- HuggingFaice (Pytorch): https://github.com/huggingface/transformers
- Tensor2Tensor (Tensorflow): https://github.com/tensorflow/tensor2tensor

Fairseq toolkit

- Simple tutorial to training your own model with Fairseq.

Step1: Preprocessing

↓

Step2: Training

↓

Step3: Decoding

Fairseq toolkit

- clone repository

```
git clone https://github.com/pytorch/fairseq
```

- install fairseq

```
cd fairseq
pip install -e ./
```

Fairseq toolkit

- Step1: Preprocessing

```
#!/bin/bash

dataPath=./dataSplit
fairseq-preprocess \
  --source-lang en --target-lang fr \
  --tokenizer moses \
  --trainpref $dataPath/train \
  --validpref $dataPath/dev \
  --destdir data-bin/tutorial-NMT
```

- source-lang: source 파일의 확장자 부분
- trainpref: train 데이터 파일의 확장자를 제외한 부분
- destdir: binary 파일이 저장될 디렉토리 경로 (없는 경우 생성)

(https://fairseq.readthedocs.io/en/latest/command_line_tools.html#fairseq-preprocess)

Fairseq toolkit

- Step2: Training

```
#!/bin/bash
export CUDA_VISIBLE_DEVICES=0

savePath=./checkpoints/tutorial-NMT
binPath=./data-bin/tutorial-NMT
fairseq-train $binPath \
  --save-dir $savePath --keep-best-checkpoints 1 --max-epoch 20 \
  --no-progress-bar --lr 0.0005 --optimizer adam --clip-norm 0.1 \
  --dropout 0.1 --max-tokens 3000 --arch lstm \
  --encoder-embed-dim 128 --encoder-bidirectional \
  --decoder-embed-dim 128 --decoder-attention True
```

(https://fairseq.readthedocs.io/en/latest/command_line_tools.html#fairseq-train)

Fairseq toolkit

- Step3: Decoding

```
> fairseq-interactive \
    ./data-bin/tutorial-NMT \
    --path ./checkpoints/tutorial/checkpoint_best.pt \
    --beam 5 --source-lang en --target-lang fr --tokenizer moses
```

(https://fairseq.readthedocs.io/en/latest/command_line_tools.html#fairseq-interactive)