Sequence to Sequence

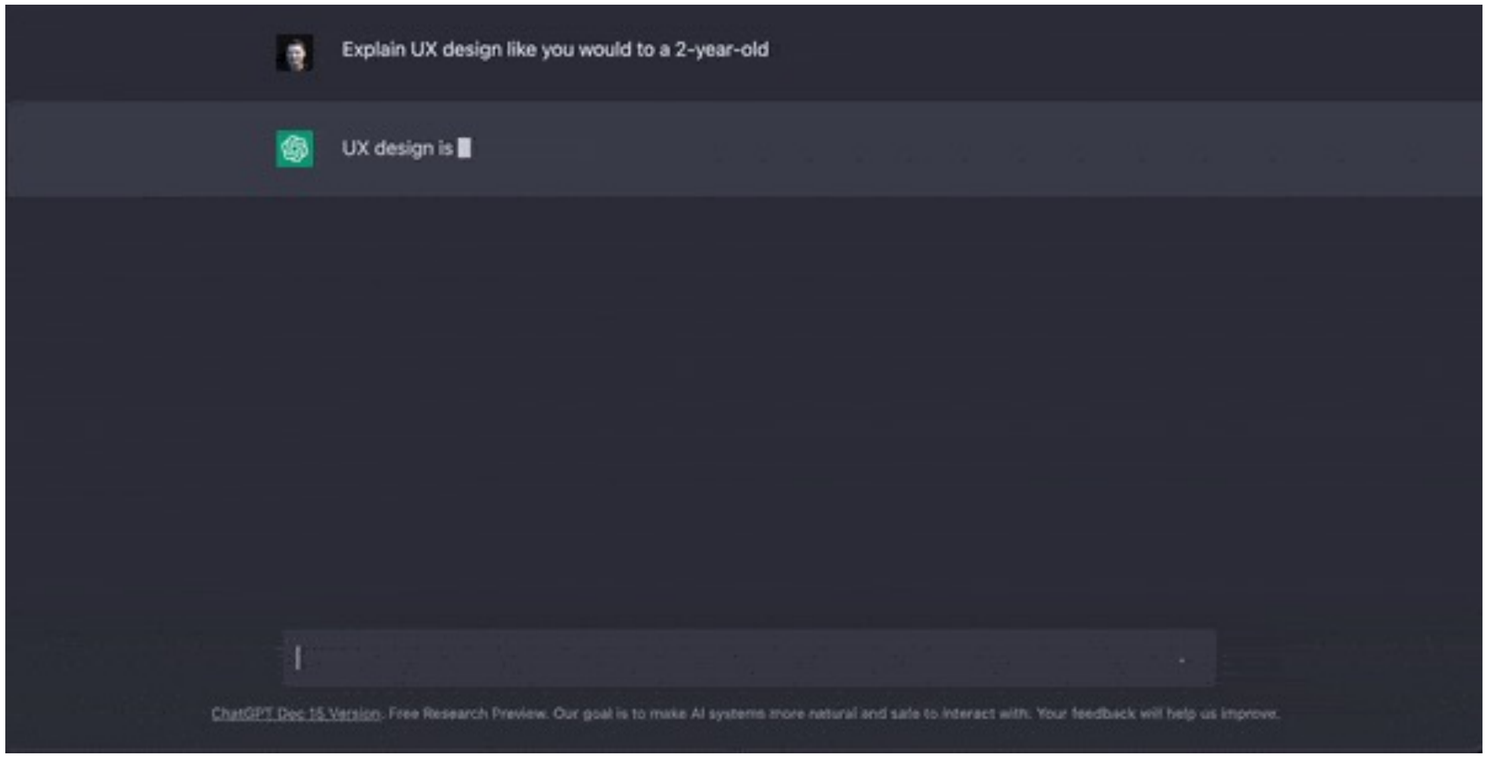
임경태

jujbob@gmail.com

Contents

생성모델의 고찰

야 그런데 ChatGPT 보니까 대화(discourse)형태로 입력 문장이 엄청 길지않냐?
 - 요구하는 입력 문장이 길고,
 - 이전에 했던 대화 문맥(Context)도 이해하고 답변하지?
우리가 배운 생성 모델은 아니자나?



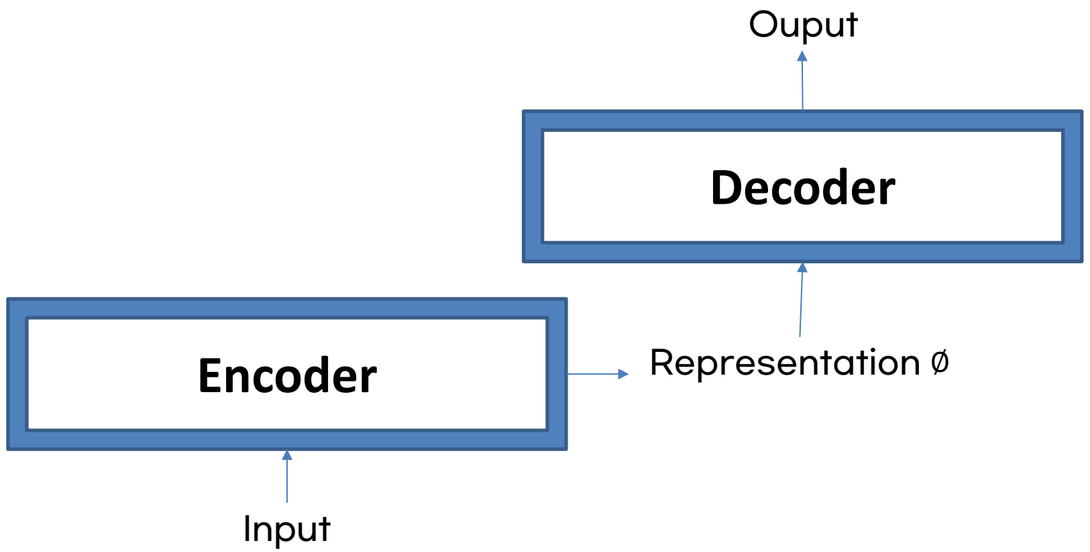
S2S 모델

Sequence-To-Sequence Modeling (Seq2Seq)

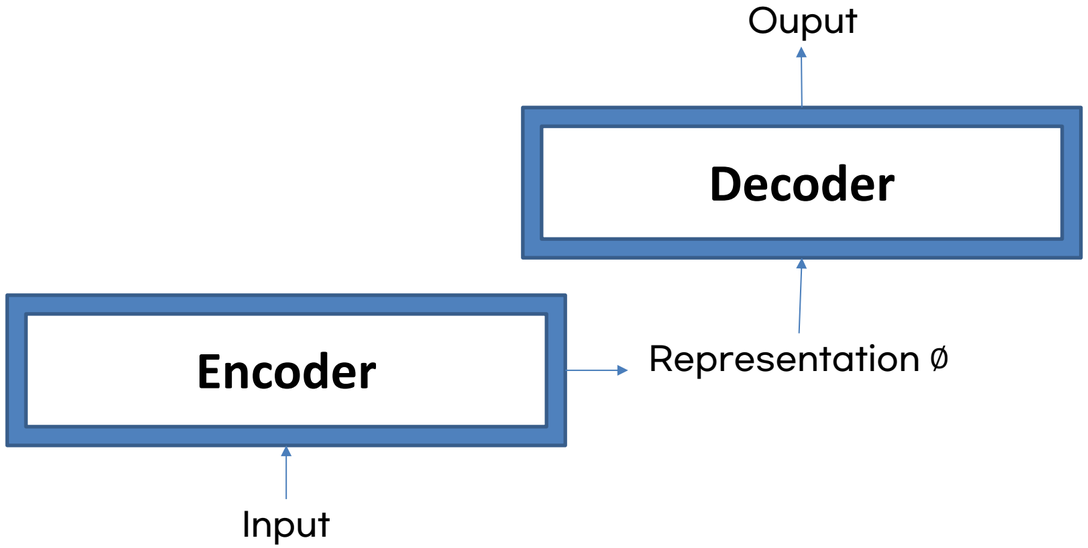
- 입력 문장을 입력 받아 출력 문장을 생성하는 모델
  - 입력과 출력 문장의 길이가 다름
  - Ex) 이메일 답장 만들기, 프랑스어 문장을 영어로 번역하기, 기사 요약하기

- 본 강의에서는
  1. seq2seq 모델의 기본 개념을 소개 변종인 양방향 모델을 소개
  2. Seq2seq를 활용한 챗봇 모델을 구현
  3. 신경망 기계 번역(netural machine translation, NMT) 구현

- S2S 모델은 Encoder - Decoder model로 구성됨



1. 인코더 모델은 입력을 받아 인코딩 혹은 표현 ø을 만듬
   - 출력 결과는 일반적으로 벡터 하나
2. 표현을 디코더 모델의 입력으로 사용해 원하는 출력을 만듬
   - 인코더와 디코더 모델은 시퀀스 모델이고 입력과 출력 둘다 시퀀스임



- 인코더 - 디코더 모델은 **조건부 생성 모델**(conditioned generation model)의 일종
- 입력 표현 ∅대신 일반적은 조건 문맥 c를 사용해 디코더가 출력을 만듬

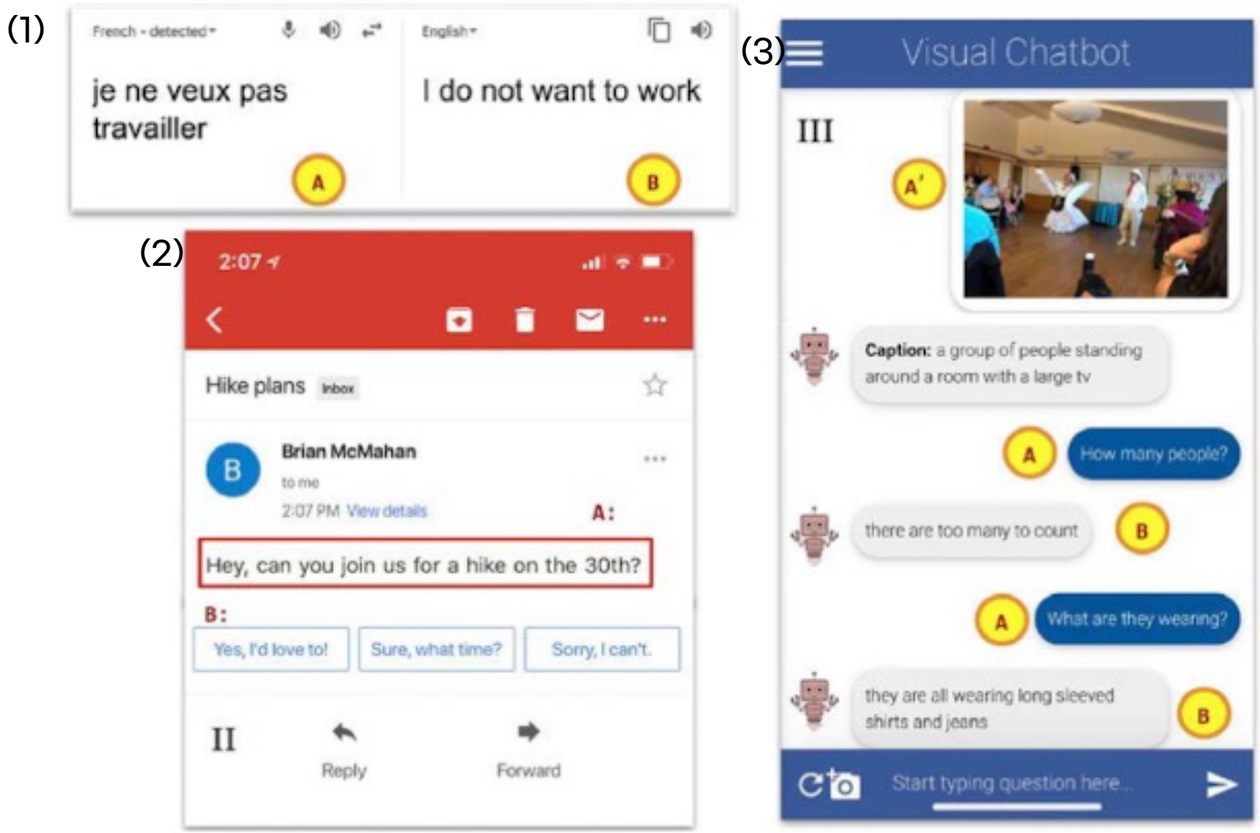
Generative Model

Generating random bag images

Conditional generative model

Generating random bag images given a sketch

https://velog.io/@ysw2946/8.-Conditional-Generative-Model



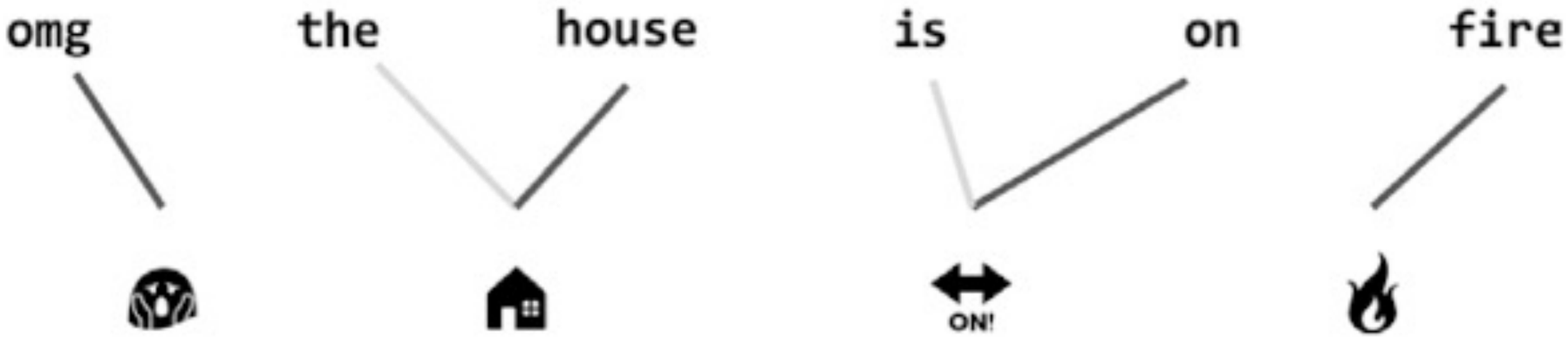
1. 기계 번역(입력 A: 프랑스어 문장, 출력 B: 영어 문장)
2. 이메일 응답 추천(입력 A: 이메일 텍스트, 출력 B: 가능한 답변 중 하나)
3. 챗봇(입력 A: 입력 이미지(A')에 관한 질문, 출력 B: 답변)

S2S 모델을 기계 번역에 이용하는 법

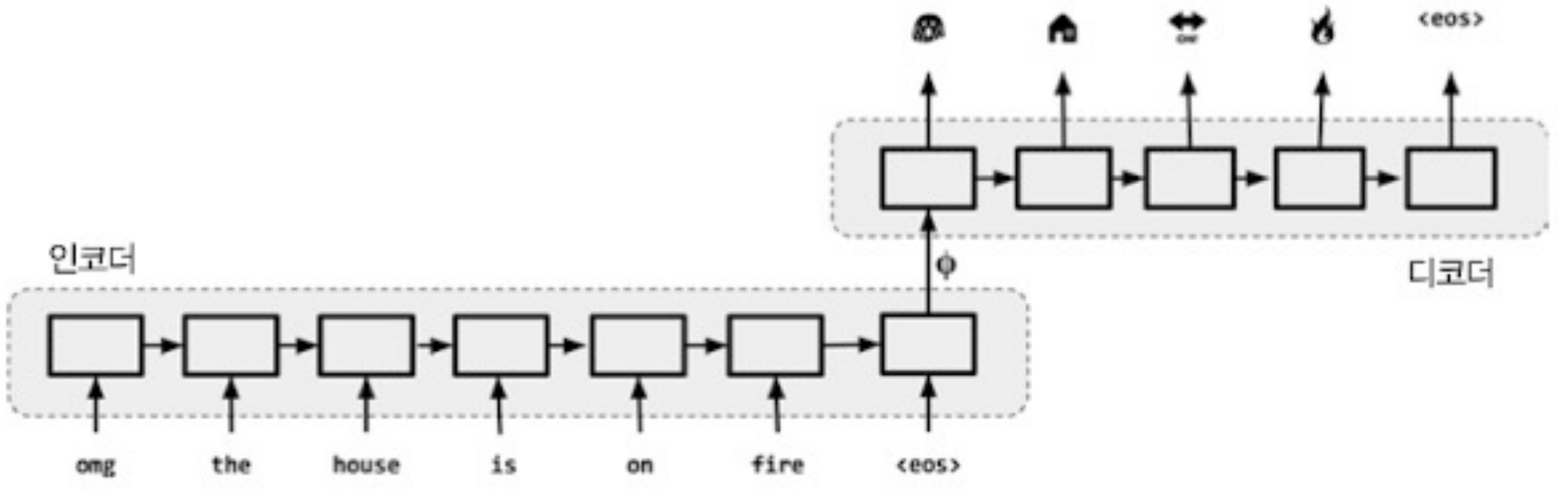
입력한 텍스트 -> emoji로 바꾸는 iOS/안드로이드 키보드

“omg the house is on fire” -> 

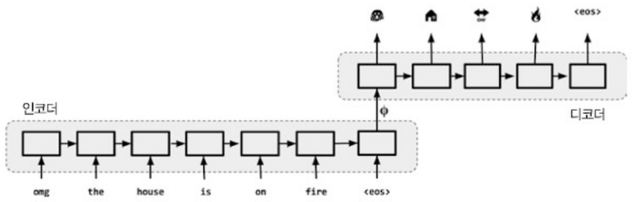
입력 토큰(6개) 출력 토큰(4)로 길이가 다름



입력과 출력 사이의 매핑을 정렬



영어를 이모지로 번역하는 S2S 모델



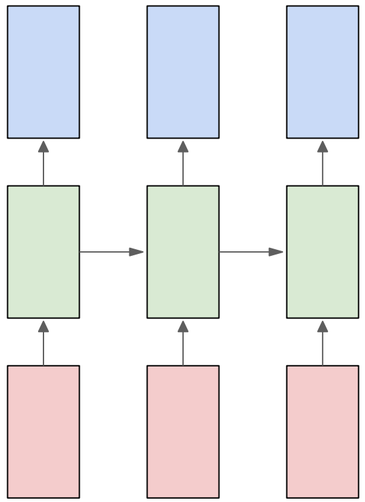
그래서 Encoder랑 Decoder는 어떻게 구현하는데?

어라? 근데 생김새가 딱 RNN모양인데?

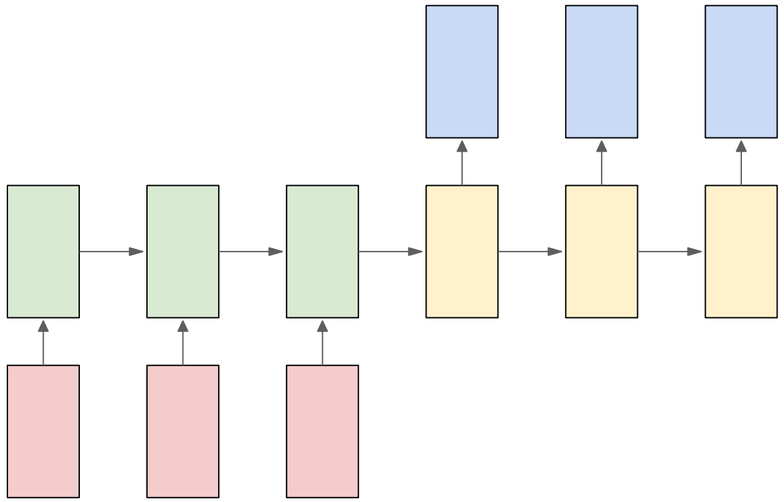
Contents

RNN 기반의 seq2seq 의 구조

- What is the difference between general RNN model and Seq2Seq model?
- We just need two different RNN structures to build seq2seq!



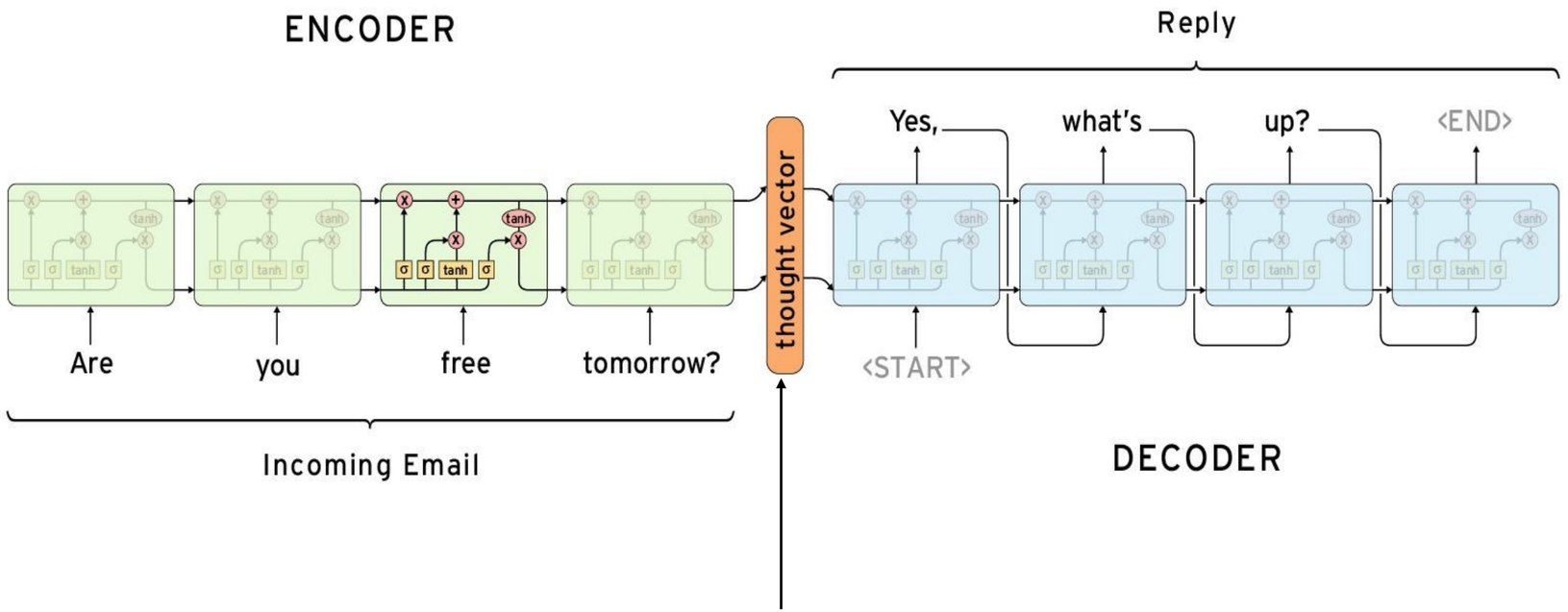
RNN



Seq2Seq

RNN 기반의 seq2seq Chatbot

- What is the difference between general RNN model and Seq2Seq model?
- We just need two different RNN structures to build seq2seq!



인코더의 hidden state를 Decoder의 첫번째 hidden state로 주입!!

https://towardsdatascience.com/sequence-to-sequence-tutorial-4fde3ee798d8

Contents

RNN 기반의 seq2seq Chatbot 구현

- **Main Function**

```
import random
import torch
import torch.nn as nn
import torch.optim as optim

SOURCE_MAX_LENGTH = 10
TARGET_MAX_LENGTH = 12
load_pairs, load_source_vocab, load_target_vocab = preprocess(raw, SOURCE_MAX_LENGTH, TARGET_MAX_LENGTH)
print(random.choice(load_pairs))

enc_hidden_size = 16
dec_hidden_size = enc_hidden_size
enc = Encoder(load_source_vocab.n_vocab, enc_hidden_size).to(device)
dec = Decoder(dec_hidden_size, load_target_vocab.n_vocab).to(device) 193
train(load_pairs, load_source_vocab, load_target_vocab, enc, dec, 5000, print_every=1000)
evaluate(load_pairs, load_source_vocab, load_target_vocab, enc, dec, TARGET_MAX_LENGTH)
```

RNN 기반의 seq2seq Chatbot 구현

- **Data Preprocessing**

```
torch.manual_seed(0)
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")

raw = ["I feel hungry.  나는 배가 고프다.",
   "Pytorch is very easy.     파이토치는 매우 쉽다.",
   "Pytorch is a framework for deep learning.     파이토치는 딥러닝을 위한 프레임워크이다.",
   "Pytorch is very clear to use.    파이토치는 사용하기 매우 직관적이다."]
SOS_token = 0
EOS_token = 1
```

RNN 기반의 seq2seq Chatbot 구현

- **Data Preprocessing**

```
43 def preprocess(corpus, source_max_length, target_max_length):
44    print("reading corpus...")
45    pairs = []
46    for line in corpus:
47       pairs.append([s for s in line.strip().lower().split("\t")])
48    print("Read {} sentence pairs".format(len(pairs))) 49
50
51    pairs = [pair for pair in pairs if filter_pair(pair, source_max_length, target_max_length)] print("Trimmed to {}
52    sentence pairs".format(len(pairs)))
53    source_vocab = Vocab()
54    target_vocab = Vocab()
55    print("Counting words...") for pair in pairs:
56       source_vocab.add_vocab(pair[0])
57       target_vocab.add_vocab(pair[1])
58    print("source vocab size =", source_vocab.n_vocab)
59    print("target vocab size =", target_vocab.n_vocab)
60
61    return pairs, source_vocab, target_vocab
62
63
64
65
```

RNN 기반의 seq2seq Chatbot 구현

- **Models**

```
66 class Encoder(nn.Module):
67   def __init__(self, input_size, hidden_size):
68     super(Encoder, self).__init__()
69     self.hidden_size = hidden_size
70     self.embedding = nn.Embedding(input_size, hidden_size)
71     self.gru = nn.GRU(hidden_size, hidden_size) 72
73
74   def forward(self, x, hidden):
75     x = self.embedding(x).view(1, 1, -
76     1) x, hidden = self.gru(x, hidden)
77     return x, hidden
78
```

RNN 기반의 seq2seq Chatbot 구현

- **Models**

```
79 class Decoder(nn.Module):
80   def __init__(self, hidden_size, output_size):
81     super(Decoder, self).__init__()
82     self.hidden_size = hidden_size
83     self.embedding = nn.Embedding(output_size,
       hidden_size)
84     self.gru = nn.GRU(hidden_size, hidden_size)
85     self.out = nn.Linear(hidden_size, output_size)
86     self.softmax = nn.LogSoftmax(dim=1) 87
88
89
90   def forward(self, x, hidden):
91     x = self.embedding(x).view(1, 1, -1)
92     x, hidden = self.gru(x, hidden)
93     x = self.softmax(self.out(x[0]))
94     return x, hidden
```

감사합니다.