KU KONKUK UNIVERSITY

Recurrent Neural Network

자연어처리연구실

PART-II

Many-to-Many Model

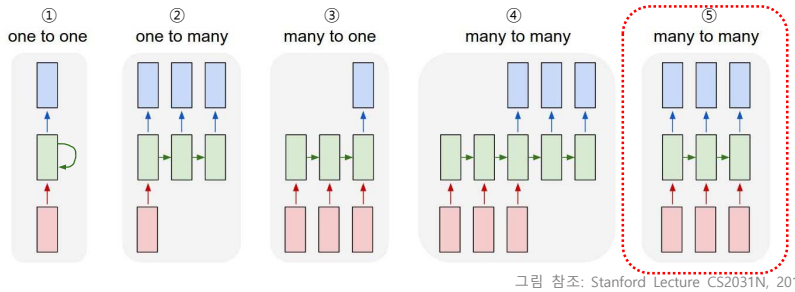


그림 참조: Stanford Lecture CS2031N, 2017
(Fei-Fei Li & Justin Johnson & Serena Yeung)

Blue : output vector
Green : RNN's state
Red : input vector

[활용 예]
① 일반적인 NN (FNN)
② 자막 생성 (image captioning)
③ 분류 및 예측 (prediction)
④ 기계 번역
⑤ 순차 표지 부착

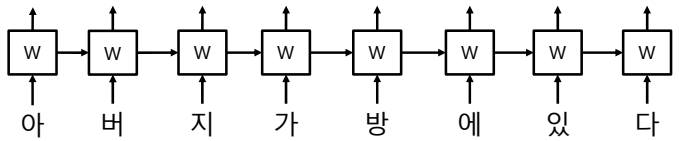
Input sequence → Label sequence
(자동 띄어쓰기, 개체명 인식 등)

Many-to-Many Model (Sequence Labeling)

- 순차적 레이블 부착
  - 연속된 입력에 대해 문맥을 반영하여 분류를 수행하는 것

BIO Notation for Segmentation: B(Beginner), I(Inner), O(outer)

개체명 인식	B-PER	I	I	O	B-LOC	O	O	O
품사 부착	B-NNG	I	I	B-JKS	B-NNG	B-JKB	B-VV	B-EC
형태소 분리	B	I	I	B	B	B	B	B
띄어쓰기	B	I	I	I	B	I	B	I
입력	아	버	지	가	방	에	있	다



실습

실습 코드 다운로드:
https://github.com
/KUNLP/Lecture

- RNN을 이용하여 자동 띄어쓰기를 수행하는 프로그램을 작성하시오.
  - 입력(관측)
    - 한글 음절
  - 출력(레이블): B, I
    - B: 관측된 음절 앞에 공백을 추가해야 함을 나타내는 레이블
    - I: 관측된 음절 앞에 공백을 추가하지 말아야 함을 나타내는 레이블

출력: B I B I I B I
입력: 나 는 사 과 가 좋 아

  - 데이터 형식
    - 한글 음절 열 ₩t 레이블 열
    - 예제: 나 는 사 과 가 좋 아 ₩t B I B I I B I

실습

모델 설계

```
class SpacingRNN(nn.Module):

    def __init__(self, config):
        super(SpacingRNN, self).__init__()

        # 전체 음절 개수
        self.eumjeol_vocab_size = config["eumjeol_vocab_size"]

        # 음절 임베딩 사이즈
        self.embedding_size = config["embedding_size"]

        # RNN 히든 사이즈
        self.hidden_size = config["hidden_size"]

        # 분류할 라벨의 개수
        self.number_of_labels = config["number_of_labels"]

        # 임베딩층: 랜덤 초기화 후 fine-tuning
        self.embedding = nn.Embedding(num_embeddings=self.eumjeol_vocab_size, embedding_dim=self.embedding_size, padding_idx=0)

        self.dropout = nn.Dropout(config["dropout"])

        # RNN layer
        #self.bi_gru = nn.GRU(input_size=self.embedding_size, hidden_size=self.hidden_size, num_layers=1, batch_first=True, bidirectional=True)
        self.bi_lstm = nn.LSTM(input_size=self.embedding_size, hidden_size=self.hidden_size, num_layers=1, batch_first=True, bidirectional=True)

        # fully_connected layer를 통하여 출력 크기를 number_of_labels에 맞춰줌
        # (batch_size, max_length, hidden_size*2) -> (batch_size, max_length, number_of_labels)
        self.linear = nn.Linear(in_features=self.hidden_size * 2, out_features=self.number_of_labels)
```


실습

모델 설계

```
def forward(self, inputs):
    # (batch_size, max_length) -> (batch_size, max_length, embedding_size)
    eumjeol_inputs = self.embedding(inputs)

    # hidden_outputs, hidden_states = self.bi_gru(eumjeol_inputs)
    hidden_outputs, hidden_states = self.bi_lstm(eumjeol_inputs)

    # (batch_size, max_length, hidden_size*2)
    hidden_outputs = self.dropout(hidden_outputs)

    # (batch_size, max_length, hidden_size*2) -> (batch_size, max_length, number_of_labels)
    hypothesis = self.linear(hidden_outputs)

    return hypothesis
```

데이터 읽기

```
def read_datas(file_path):
    with open(file_path, "r", encoding="utf8") as inFile:
        lines = inFile.readlines()
    datas = []
    for line in lines:
        # 입력 문장을 ₩t으로 분리
        pieces = line.strip().split("₩t")
        # 입력 문자열을 음절 단위로 분리
        eumjeol_sequence, label_sequence = pieces[0].split(), pieces[1].split()
        datas.append((eumjeol_sequence, label_sequence))
    return datas
```


실습

사전 읽기

```
def read_vocab_data(eumjeol_vocab_data_path):
    label2idx, idx2label = {"<PAD>":0, "B":1, "I":2}, {0:"<PAD>", 1:"B", 2:"I"}
    eumjeol2idx, idx2eumjeol = {}, {}

    with open(eumjeol_vocab_data_path, "r", encoding="utf8") as inFile:
        lines = inFile.readlines()

    for line in lines:
        eumjeol = line.strip()
        eumjeol2idx[eumjeol] = len(eumjeol2idx)
        idx2eumjeol[eumjeol2idx[eumjeol]] = eumjeol

    return eumjeol2idx, idx2eumjeol, label2idx, idx2label
```

eumjeol_vocab.txt - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
나
는
사
과
가
좋
아

```
def load_dataset(config):
    datas = read_datas(config["input_data"])
    eumjeol2idx, idx2eumjeol, label2idx, idx2label = read_vocab_data(config["eumjeol_vocab"])

    for eumjeol_sequence, label_sequence in datas:
        eumjeol_feature = [eumjeol2idx[eumjeol] for eumjeol in eumjeol_sequence]
        label_feature = [label2idx[label] for label in label_sequence]

        # 음절 sequence의 실제 길이
        eumjeol_feature_length = len(eumjeol_feature)

        # 모든 입력 데이터를 고정된 길이로 맞춰주기 위한 padding 처리
        eumjeol_feature += [0] * (config["max_length"] - eumjeol_feature_length)
        label_feature += [0] * (config["max_length"] - eumjeol_feature_length)

    return eumjeol_features,
    eumjeol_feature_lengths,
    label_features, eumjeol2idx,
    idx2eumjeol, label2idx, idx2label
```

실습

Train

```
def train(config):
    # RNN 모델 객체 생성
    model = SpacingRNN(config).cuda()

    # 데이터 읽기
    eumjeol_features, eumjeol_feature_lengths, label_features, eumjeol2idx, idx2eumjeol, label2idx, idx2label = load_dataset(config)
```

```
    for epoch in range(config["epoch"]):

        model.train()
        costs = []

        for step, batch in enumerate(train_dataloader):
```

```
            # 음절 데이터, 각 데이터의 실제 길이, 라벨 데이터
            inputs, input_lengths, labels = batch[0], batch[1], batch[2]

            # 모델 출력 결과 얻어오기
            hypothesis = model(inputs)

            # hypothesis : (batch_size, max_length, number_of_labels) -> (batch_size*max_length, number_of_labels)
            # labels : (batch_size, max_length) -> (batch_size*max_length, )
            cost = loss_func(hypothesis.reshape(-1, len(label2idx)), labels.flatten())
            cost.backward()
            optimizer.step()
```

Edited by Harksoo Kim

실습

Test

```
def test(config):
    # 데이터 읽기
    eumjeol_features, eumjeol_feature_lengths, label_features, eumjeol2idx, idx2eumjeol, label2idx, idx2label = load_dataset(config)

    # RNN 모델 객체 생성
    model = SpacingRNN(config).cuda()
    # 사전학습한 모델 파일로부터 가중치 불러옴
    model.load_state_dict(torch.load(os.path.join(config["output_dir_path"], config["model_name"])))

    for step, batch in enumerate(test_dataloader):

        # 음절 데이터, 각 데이터의 실제 길이, 라벨 데이터
        inputs, input_lengths, labels = batch[0], batch[1], batch[2]

        # 모델 평가
        hypothesis = model(inputs)

        # (batch_size, max_length, number_of_labels) -> (batch_size, max_length)
        hypothesis = torch.argmax(hypothesis, dim=-1)

        # batch_size가 1이기 때문
        input_length = tensor2list(input_lengths[0])
        input = tensor2list(inputs[0])[:input_length]
        label = tensor2list(labels[0])[:input_length]
        hypothesis = tensor2list(hypothesis[0])[:input_length]

        # 출력 결과와 정답을 리스트에 저장
        total_hypothesis += hypothesis
        total_labels += label
```

Edited by Harksoo Kim

실습

Test

```
if (step < 10):
    # 정답과 모델 출력 비교
    predict_sentence, correct_sentence = make_sentence(
    print("정답 : " + correct_sentence)
    print("출력 : " + predict_sentence)
    print()
```

```
def make_sentence(inputs, predicts, labels, idx2eumjeol, idx2label):

    predict_sentence, correct_sentence = "", ""

    for index in range(len(inputs)):
        eumjeol = idx2eumjeol[inputs[index]]
        correct_label = idx2label[labels[index]]
        predict_label = idx2label[predicts[index]]

        # 시작 음절인 경우 공백을 추가해줄 필요가 없음
        if (index == 0):
            predict_sentence += eumjeol
            correct_sentence += eumjeol
            continue

        # "B" 태그인 경우 어절의 시작 음절이므로 앞에 공백을 추가
        if (predict_label == "B"):
            predict_sentence += " "
        predict_sentence += eumjeol

        # "B" 태그인 경우 어절의 시작 음절이므로 앞에 공백을 추가
        if (correct_label == "B"):
            correct_sentence += " "
        correct_sentence += eumjeol

    return predict_sentence, correct_sentence
```

Edited by Harksoo Kim

실습

Main

```
if(__name__=="__main__"):
    root_dir = "/gdrive/My Drive/colab/rnn/spacing"
    output_dir = os.path.join(root_dir, "output")
    if not os.path.exists(output_dir):
        os.makedirs(output_dir)

    config = {"mode": "test",
              "model_name":"epoch_{0:d}.pt".format(5),
              "input_data":os.path.join(root_dir, "train.txt"),
              "output_dir_path":output_dir,
              "eumjeol_vocab": os.path.join(root_dir, "eumjeol_vocab.txt"),
              "label_vocab": os.path.join(root_dir, "label_vocab.txt"),
              "eumjeol_vocab_size": 2458,
              "embedding_size": 100,
              "hidden_size": 100,
              "max_length": 920,
              "number_of_labels": 3,
              "epoch":5,
              "batch_size":64,
              "dropout":0.3
              }

    if(config["mode"] == "train"):
        train(config)
    else:
        test(config)
```

```
Average cost : 0.5464620601527298
Average cost : 0.27692207882675945
Average cost : 0.2222117374214945
Average cost : 0.19301371408414236
Average cost : 0.1738766309203981
```

```
정답 : 부인이 정성들여 키운 진이 독살되었다는 것은 안
출력 : 부인이 정성들여키운진이 독살되었다는 것은 연부

정답 : 이같은 그의 가난은 어려서만이 아니라 자란 뒤에
출력 : 이 같은 그의 가난은 어려서만이 아니라 자란 뒤

정답 : 어느 물리학자의 머리 속.
출력 : 어느 물리학자의 머리속.
```

Edited by Harksoo Kim

Many-to-Many Model (Language Model)

- 언어 모델
  - 대표적인 sequence generation 과업
  - 학습 말뭉치에서 언어(문장)가 생성될 확률을 모델링한 것
  - 선택된 단어를 다음 입력으로 사용하는 방법을 문장이 완성될 때까지 반복 → 단어 열을 생성

$$p(tom \mid \langle s \rangle) \times p(likes \mid \langle s \rangle, tom)$$
$$\times p(beer \mid \langle s \rangle, tom, likes)$$
$$\times p(\langle /s \rangle \mid \langle s \rangle, tom, likes, beer)$$

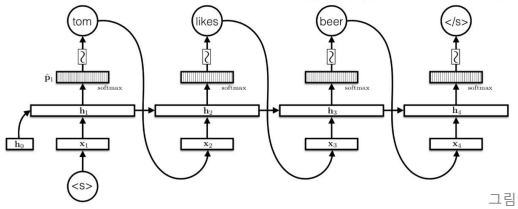


그림 출처: Chris Dyer 교수 강의자료

Many-to-Many Model (Sequence-to-Sequence)

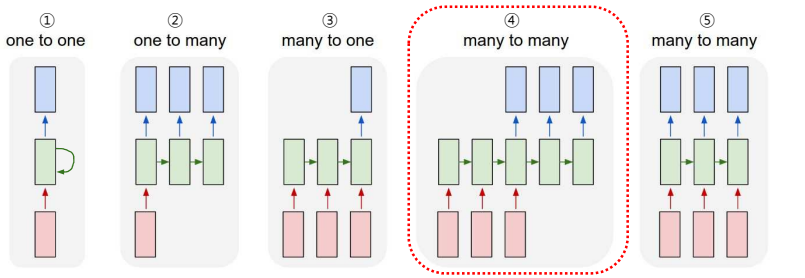


그림 참조: Stanford Lecture CS2031N, 2017
(Fei-Fei Li & Justin Johnson & Serena Yeung)

Blue : output vector
Green : RNN's state
Red : input vector

[활용 예]
① 일반적인 NN (FNN)
② 자막 생성 (image captioning)
③ 분류 및 예측 (prediction)
④ 기계 번역
⑤ 순차 표지 부착

Korean sentence (word sequence)→
English sentence (word sequence)

Seq2Seq for Machine Translation

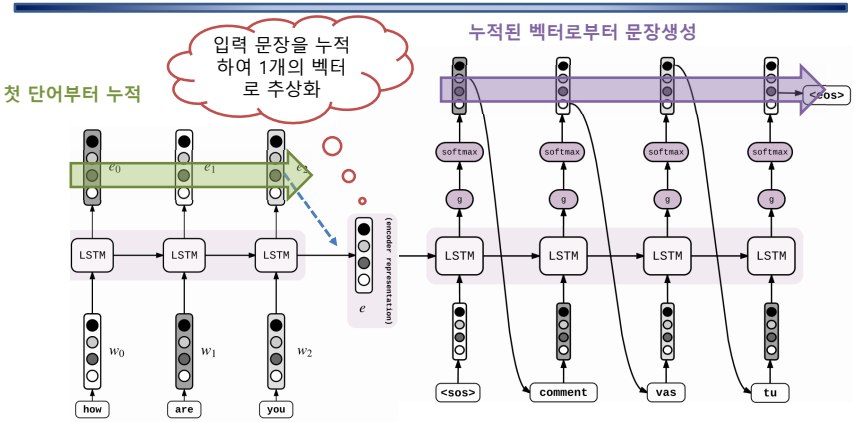


그림 출처: Guillaume Genthial 발표자료

Seq2Seq with Attention

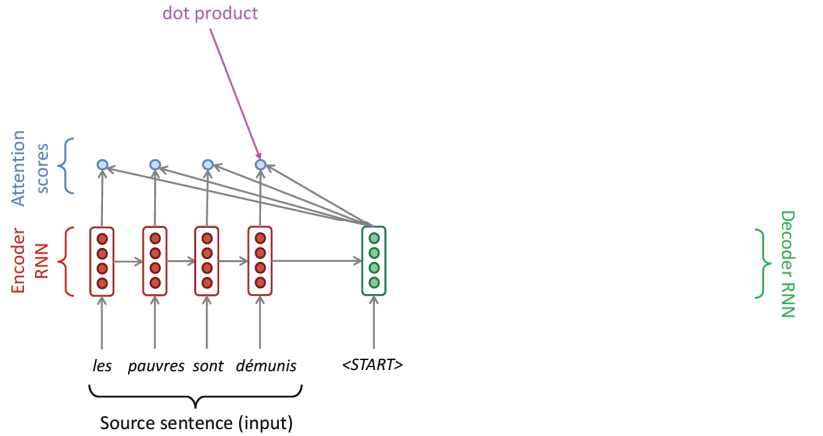


그림 출처: Stanford CS 224N/Ling 284 강의자료

Seq2Seq with Attention

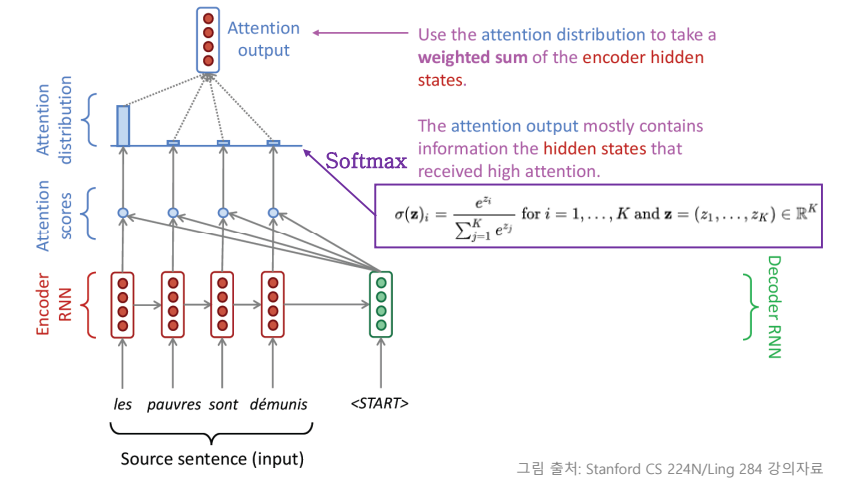


그림 출처: Stanford CS 224N/Ling 284 강의자료

Seq2Seq with Attention

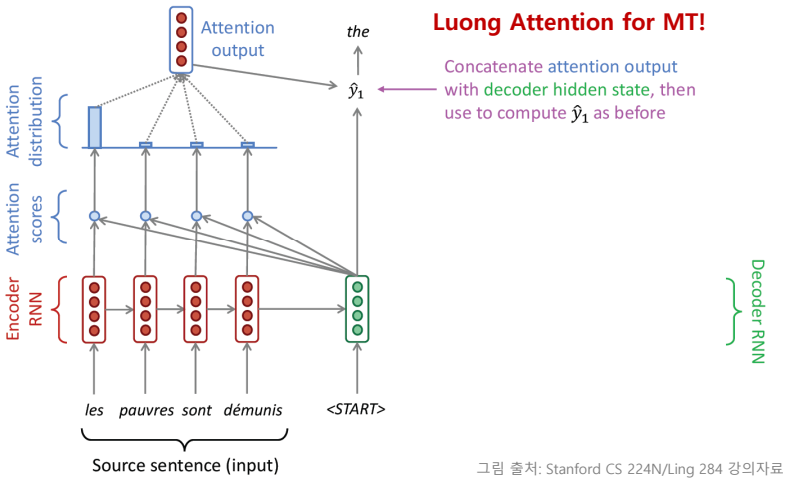
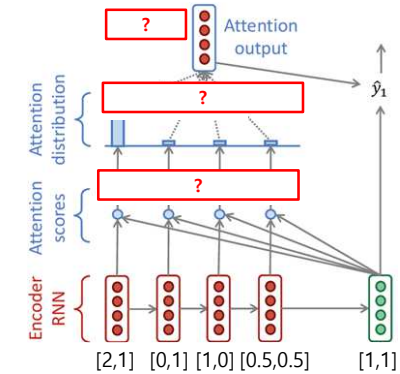


그림 출처: Stanford CS 224N/Ling 284 강의자료

확인 문제

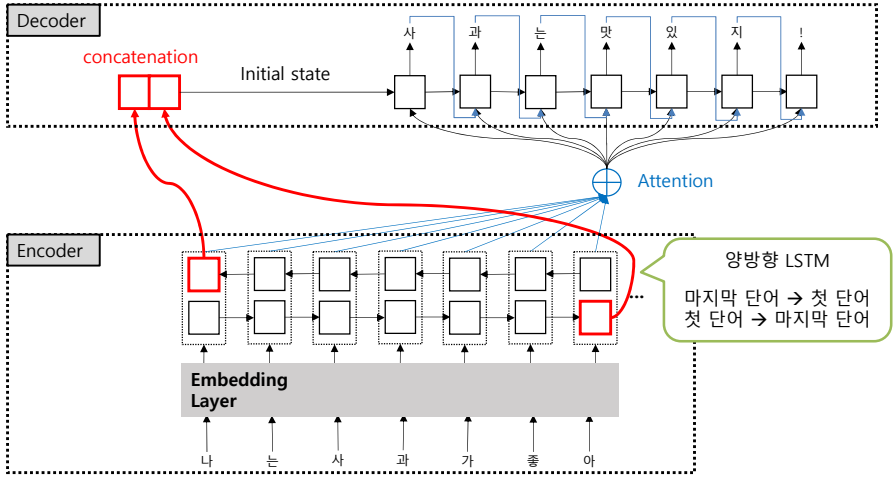
- Luong attention 수식에 기초하여 attention scores, attention distribution, attention output을 계산 하시오. (소수점 이하 2자리 반올림)



Softmax

$$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \text{ for } i = 1, \ldots, K \text{ and } \mathbf{z} = (z_1, \ldots, z_K) \in \mathbb{R}^K$$

Seq2Seq for ChatBot



시연 영상

```
I tensorflow/core/common_runtime/gpu/gpu_device.cc:972] D
I tensorflow/core/common_runtime/gpu/gpu_device.cc:982] 0
I tensorflow/core/common_runtime/gpu/gpu_device.cc:1041]
Reading model parameters from ./model/model.ckpt-1001187
Success Load!
# NLU Load Success!
I tensorflow/core/common_runtime/gpu/gpu_device.cc:975] C
Input :
```


질의응답

Q&A

Homepage: http://nlp.konkuk.ac.kr
E-mail: nlpdrkim@konkuk.ac.kr