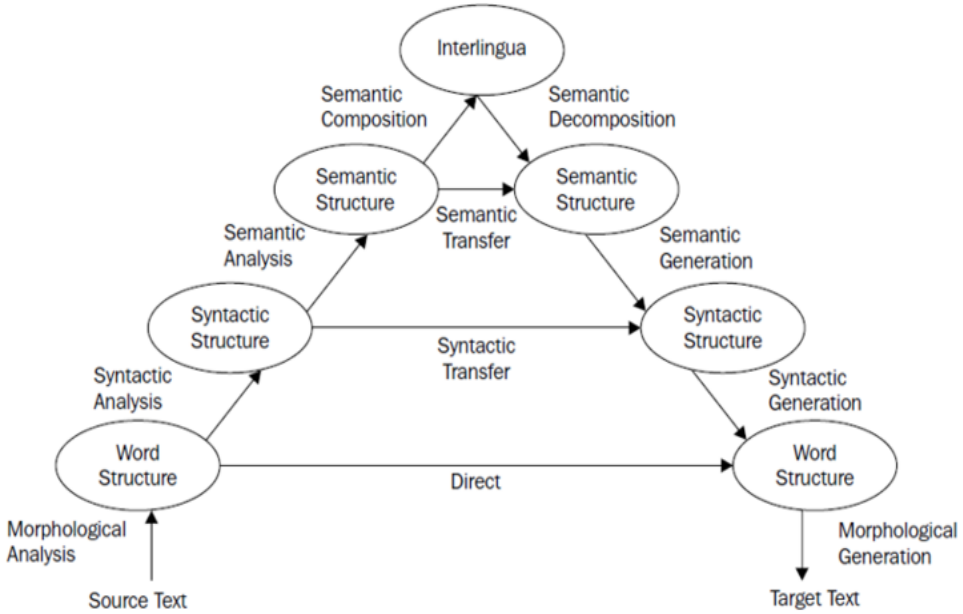
Neural Machine Translation by jointly learning to align and translate

DA 문구영, 윤준현

Contents

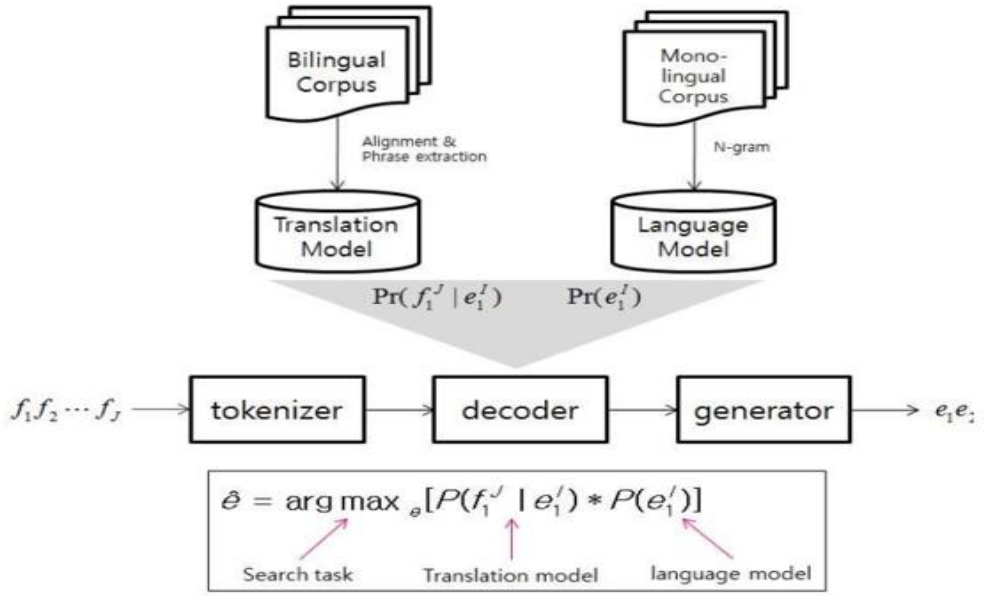
Background : Machine Translation

규칙 기반 기계 번역 (RBMT)



- 두 언어의 **문법 규칙, 언어학적 요소**를 기반으로 기계번역기 구축
- 형태 분석 → 구문 분석 → 의미 합성 → 중간 언어 → 타깃 단어
- 언어학적 지식이 많이 필요, 문법 규칙 추출이 어려움

통계적 기계 번역 (SMT)

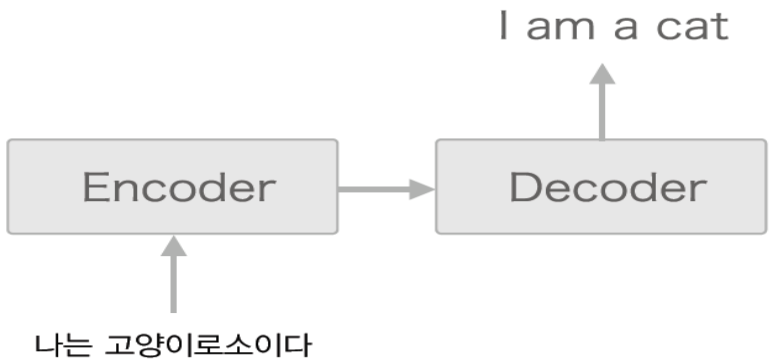


- ‘언어학자 한 명을 해고할 때마다 번역의 정확도가 높아진다‘
- **어떤 단어가 특정 단어로 번역될 통계적 확률 분포를 최대화**
- 병렬 코퍼스 필요 (원본-번역문)

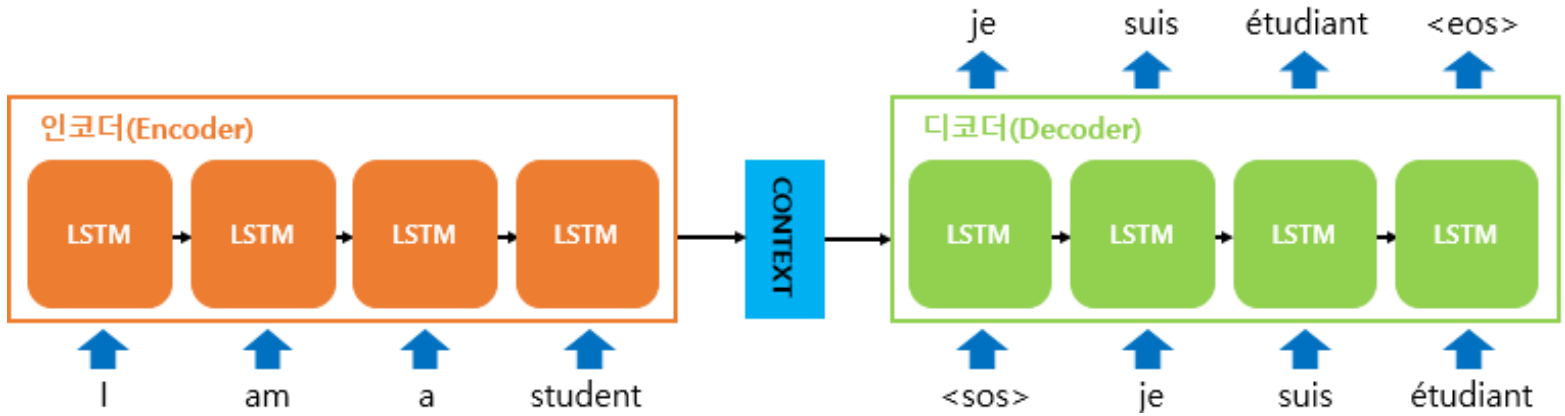
신경망 기계 번역 (NMT)

- 시계열 데이터를 다른 시계열 데이터로 변환하는 모델 (원본 문장 → 번역 문장)
- seq2seq, Encoder – Decoder

그림 7-5 Encoder와 Decoder가 번역을 수행하는 예



1. Encoder가 출발어 문장을 인코딩 (정보를 어떤 규칙에 따라 변환)
2. Encoder의 출력은 번역에 필요한 정보가 조밀하게 응축되어 있음
   * **고정된 길이의 문맥 벡터** 생성
3. Decoder는 해당 정보를 바탕으로 도착어 문장을 생성



- Encoder의 출력 벡터는 Encoder의 LSTM 계층의 마지막 은닉 상태 (고정된 길이의 문맥 벡터)

Background : RNN Encoder-Decoder

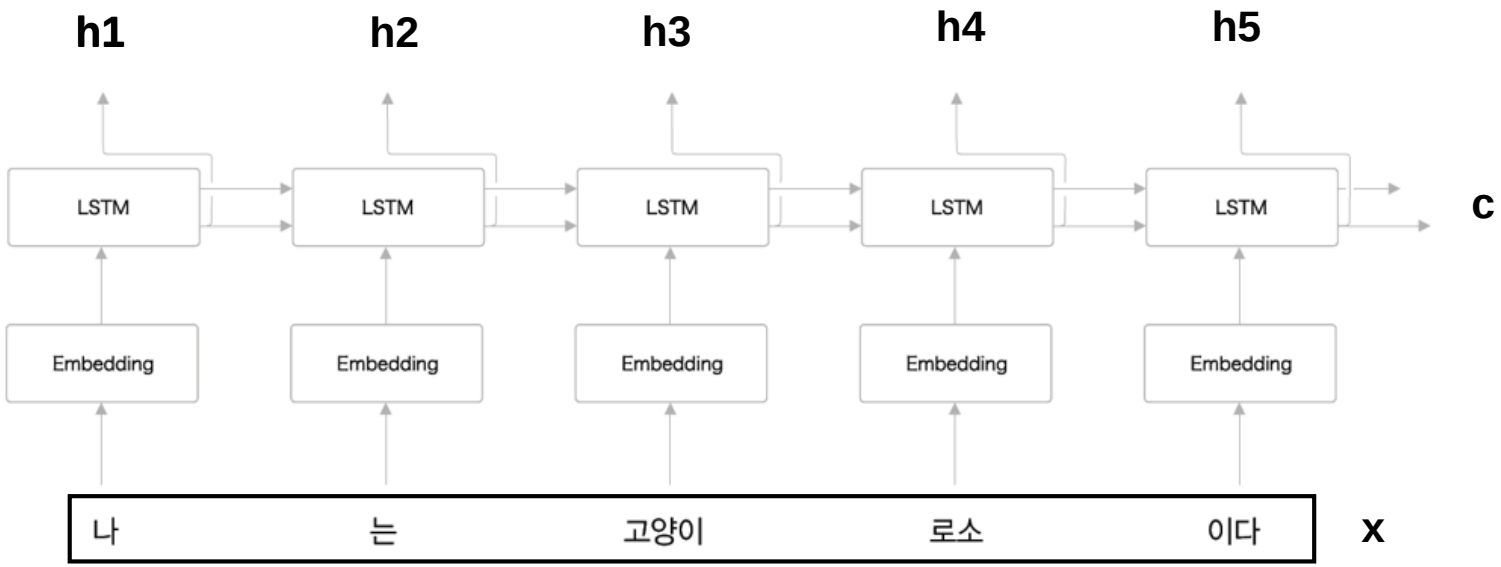
Encoder

$\mathbf{x} = (x_1, \cdots, x_{T_x}),$ → 단어(토큰) 벡터들의 sequence로 표현된 입력 문장

$h_t = f(x_t, h_{t-1})$ → 특정 time step t에서의 은닉 상태 출력 (t 시점의 단어, t-1시점의 은닉 상태 출력를 입력으로 받음)

$c = q(\{h_1, \cdots, h_{T_x}\}),$ → 문맥 벡터 (마지막 RNN 계층의 최종 은닉 상태 출력)

* f, q : 비선형 활성 함수 / **Encoder는 입력 X를 c로 mapping**하는 것이 목적



Decoder

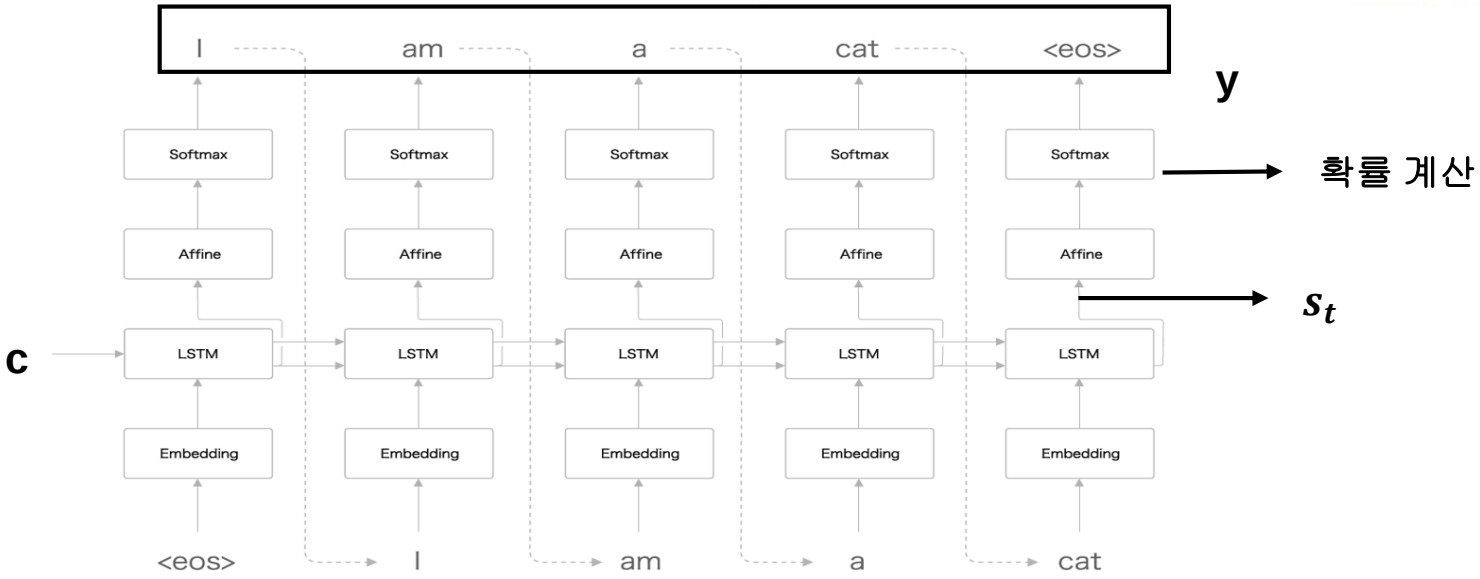
- **t 시점에 등장할 단어를 예측** (문맥 벡터 c, t-1 시점까지 예측된 단어들, t 시점의 hidden state출력이 주어졌을 때)

$$p(y_t \mid \{y_1, \cdots, y_{t-1}\}, c) = g(y_{t-1}, s_t, c),$$ → g : 특정 시점에 등장할 단어의 확률을 예측하는 비선형 네트워크

$$\mathbf{y} = (y_1, \cdots, y_{T_y}).$$ → 출력 문장 (도착어, 번역 문장)

$$p(\mathbf{y}) = \prod_{t=1}^{T} p(y_t \mid \{y_1, \cdots, y_{t-1}\}, c),$$

$P(y_1|c) \cdot P(y_2|y_1, c) \cdot P(y_3|y_2, y_1, c) \cdot \cdots \cdot P(y_t|y_1, y_2, \cdots, y_{t-1}, c)$



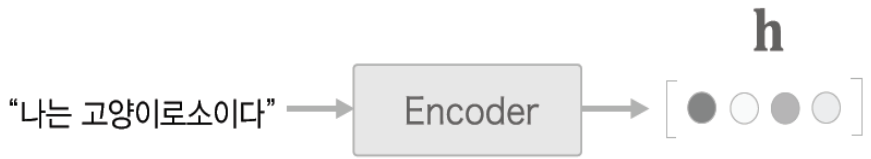
매 시점 마다 공통된 문맥 벡터를 사용하여 단어 예측

Seq2seq의 한계

- 순환 신경망을 활용하여 **고정된 길이의 context vector**에 소스 문장의 정보를 압축
- 하나의 문맥 벡터가 소스 문장의 모든 정보를 가지고 있어야 함
- 입력 문장의 길이에 관계 없이 정보를 고정된 길이의 벡터로 밀어넣음

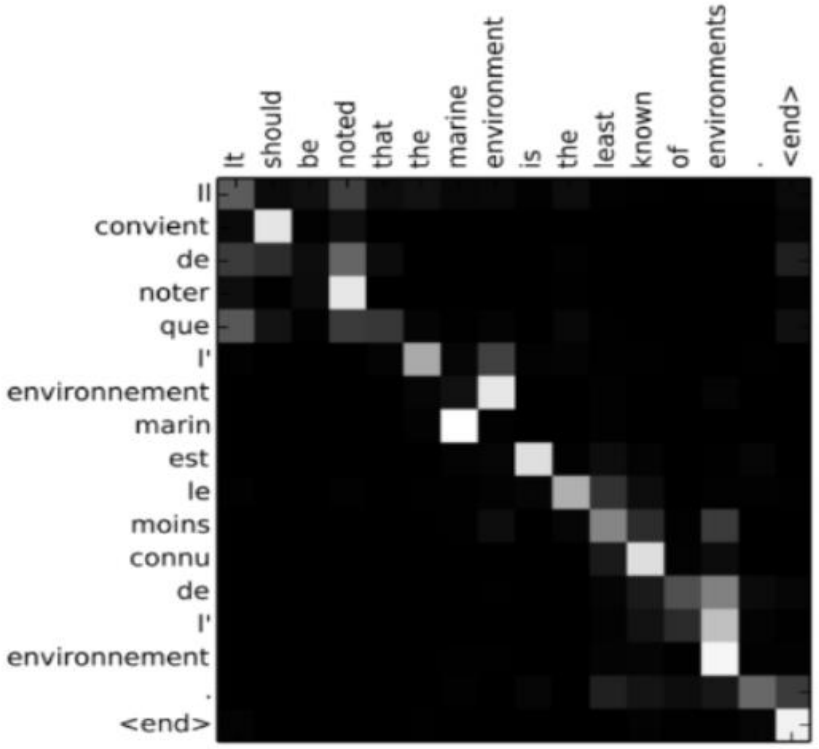
→ 입력 문장의 길이가 길어지면 **병목 현상(bottleneck) 발생, 기울기 소실 문제**

그림 8-1 입력 문장의 길이에 관계없이, Encoder는 정보를 고정 길이의 벡터로 밀어 넣는다.



대응 관계 (Alignment)

- 번역 : 나 = I, 고양이 = cat이라는 지식을 학습
- 특정한 단어에 **주목(attention)**하여 그 단어의 변환을 수시로 적용
- 입력과 출력의 여러 단어 중 **어떤 단어끼리 서로 관련되어 있는가** 라는 대응 관계의 학습을 seq2seq에게 **자동**으로 도입



- **각 출력(번역)이 어떤 입력(원문)을 얼마나 참고했는지** 표시
- 단어와 단어의 **관련성**이라는 인간이 이해할 수 있는 구조와 의미를 모델에 제공
- Soft alignment – 단어간의 일대다 관계를 표현

  ↔ hard alignment – 한 단어가 다른 한 단어와 무조건 1:1로 mapping

Bahdanau Attention - Decoder

$$p(y_i|y_1,\dots,y_{i-1},\mathbf{x}) = g(y_{i-1},s_i,c_i),$$ → 공통된 문맥 벡터 c가 아닌 특정 시점에서 예측하고자 하는 단어의 **고유한** 문맥 벡터

The context vector c_i → Encoder의 hidden states(annotations)에 의존함

$(h_1,\cdots,h_{T_x})$ → Annotations (encoder's hidden states) / 특정 i 번째 단어 주변 부분들에 더 많은 영향을 받는 정보를 담고 있음

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x}\exp(e_{ik})},$$ → Attention score : 특정 i 시점의 단어를 예측하는데 있어 Encoder의 j 번째 hidden state출력이 얼마나 중요한 지, 얼마나 주목해야 하는지 표현 / softmax 사용

$e_{ij} = a(s_{i-1},h_j)$ → Energy : Decoder의 i 시점에 입력으로 들어오는 i-1번째 시점의 hidden state출력과 Encoder의 j번째 annotation 출력간의 **연관성**
(s_{i-1} : Decoder의 i-1 시점의 hidden state 출력, i 시점의 입력)

Alignment Model

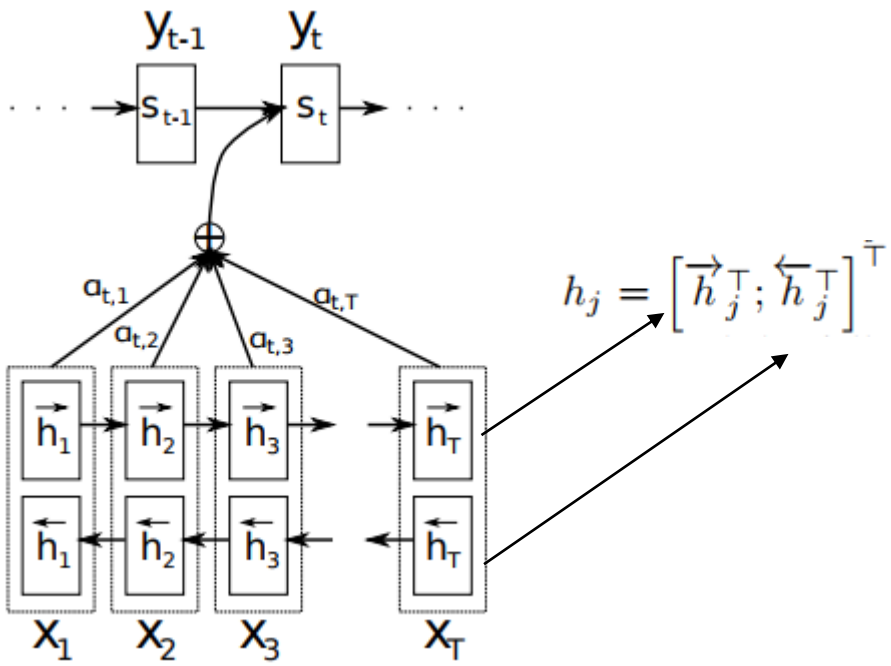
$$c_i = \sum_{j=1}^{T_x}\alpha_{ij}h_j.$$ → time step i의 고유한 문맥 벡터

Encoder : Bidirectional RNN for Annotating Sequences

- **양방향 RNN** 사용

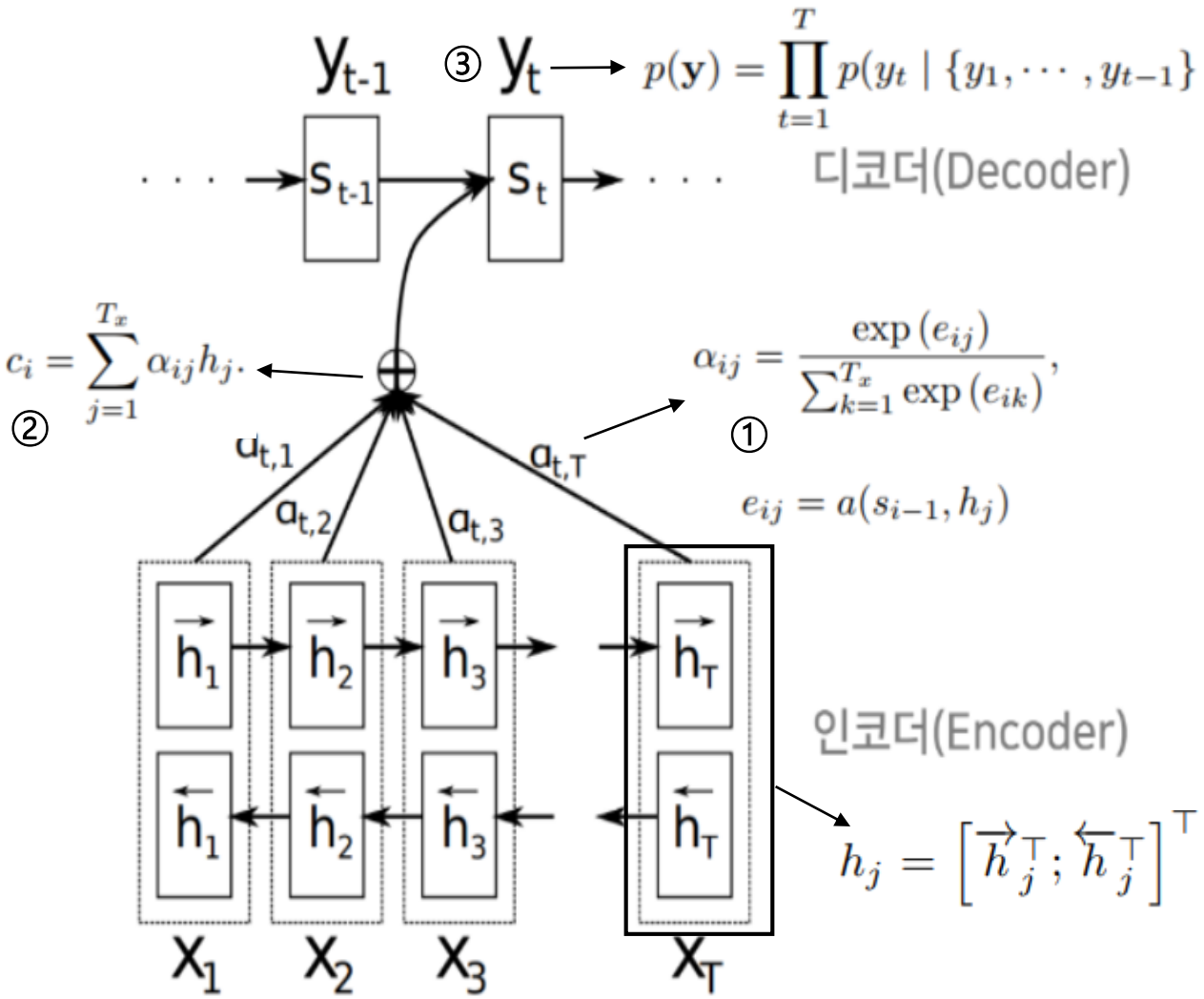
→ 입력 문장에 대해 이전에 나타나는 내용, 이후에 나타나는 내용 모두를 알아야 높은 성능의 번역 가능

→ 기울기 소실 문제에 대응



- Concatenating forward hidden state & backward hidden state
- 인접한 state의 정보를 더 많이 가지고 있는 RNN 특성상 h_j 는 단어 x_j 와 가까운 위치의 단어 정보를 더 많이 보유

Background : Machine Translation



Encoder

1. 양방향 RNN을 통해 hidden state 생성

Decoder

① 모든 Encoder의 hidden state와 Decoder가 예측하고자 하는 시점의 입력 hidden state의 **연관성** 계산 (attention score, 가중치)

② 위에서 구한 가중치를 이용하여 annotation의 가중 평균 계산 (context vector)

③ Context vector, Decoder의 hidden state vector를 활용하여 단어를 예측

Decoder

1. Encoder-Decoder attention
   → Decoder에서 하나의 결과를 생성(번역)할 때 마다, **입력 문장을 순차적으로 탐색**
   → 현재 예측하고자 하는 타깃 단어 예측에 있어 **어떤 부분들이 중요한지 판단 (attention score)**
2. Encoder에서 생성한 문맥 벡터에서 **관련성이 크다고 판단되는** (attention score가 높은) 토큰들을 활용하여 **context vector** 생성
3. Context vector, 이전 시점까지 예측한 단어들, Decoder에서 이미 생성한 hidden state vector를 기반으로 **다음 단어를 예측**

- Decoder에서 연산을 진행하며 Encoder에서 생성한 문맥 벡터를 계속해서 참조
- 예측하고자 하는 단어와 관련성이 높은 부분들에 집중

Dataset

- WMT'14의 English-French parallel corpus 사용(monolingual corpus 사용 배제)
- 전체 corpus의 단어를 348M 개로 제한(Data Selection)
- 전체 단어 중 가장 많이 사용되는 30,000개의 단어를 모델 training에 사용

  → 30,000개 외의 단어는 UNK(Unknown) 토큰으로 처리

Model

- RNNencdec(기존RNN 모델)와 RNNsearch (RNN+Attention) 2가지 모델 학습
  → 문장을 구성하는 최다 단어의 수에 따라 RNNencdec30/50으로 명명
- minibatch SGD와 Adadelta 사용(minibatch: 80문장), 5일간 학습
- 모델 성능 측정은 BLEU Score 사용

※ BLEU Score: IBM 社에서 만든 각 n-gram별 precision의 가중평균, 기계 번역 성능 측정에 사용되는 주요지표

$$\text{BLEU}(\hat{y}, y) = \text{brevity\_penalty}(\hat{y}, y) \times \prod_{n=1}^{N} p_n^{w_n},$$

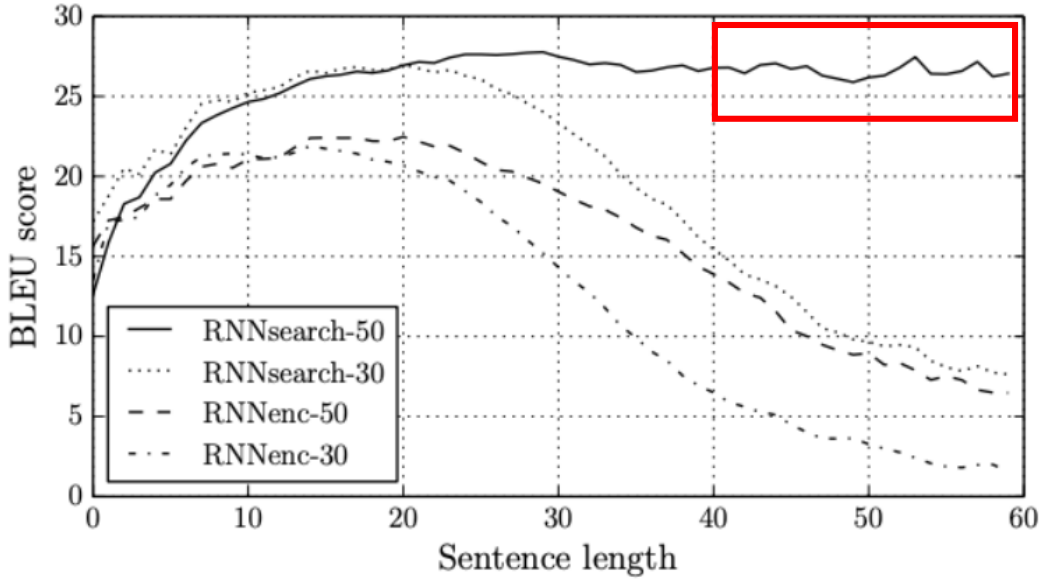
$$\text{where brevity\_penalty}(\hat{y}, y) = \min\left(1, \frac{|\hat{y}|}{|y|}\right)$$

$$\text{and } p_n^{w_n} \text{ is precision of } n\text{-gram with weight } w_n = \frac{1}{2^n}.$$

Result (Quantitative Research)

- RNNsearch가 동일 문장 길이 모델(30/50)에서 RNNencdec보다 더 높은 BLEU Score를 기록함
  → 특히, **RNNsearch-50은 문장 길이가 길어져도 높은 BLEU Score를 유지**
- RNNsearch-50*는 improvement가 더 이상 없을 때까지 학습시킨 모델을 의미하며,
  Moses는 Phrase-based 방식의 NMT로, 별도의 monolingual corpus로 추가 학습까지 진행
  → UNK 토큰을 제거한 데이터에서는 RNNsearch-50*의 BLEU Score가 가장 높게 나옴(36.15)

Model	All	No UNK°
RNNencdec-30	13.93	24.19
RNNsearch-30	21.50	31.44
RNNencdec-50	17.82	26.71
RNNsearch-50	26.75	34.16
RNNsearch-50*	28.45	36.15
Moses	33.30	35.63

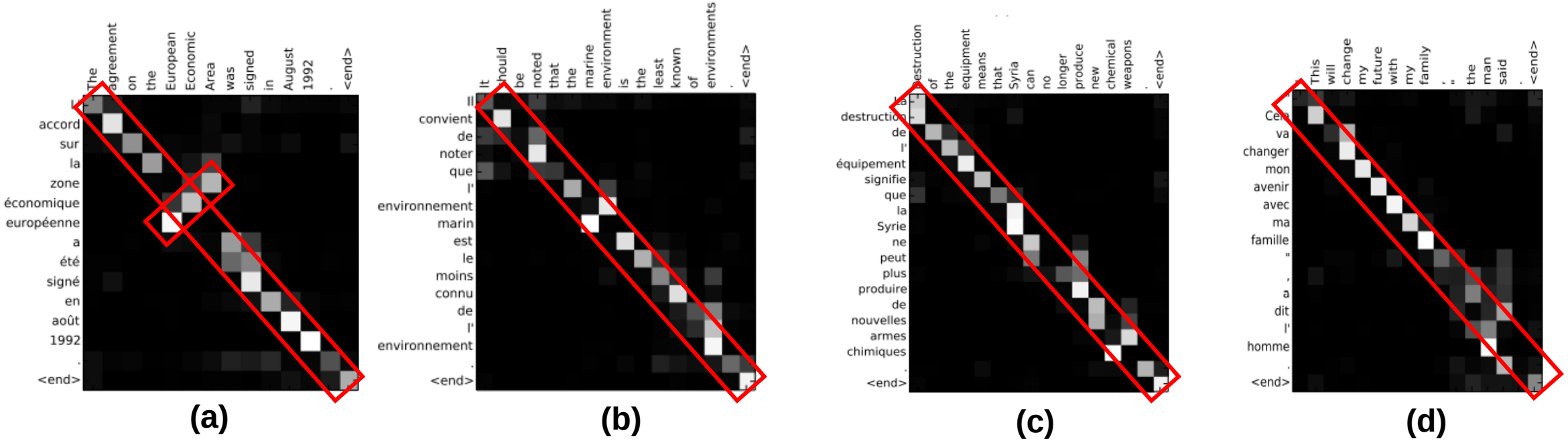


Result (Qualitative Research)

- RNNsearch-50을 대상으로 영어와 불어 사이의 alignment model에 대한 matrix를 표현해본 결과,
  영어와 불어의 alignment는 대체로 단조롭게 나타남 (그림 (a)처럼 단조롭지 않은 부분도 존재)
- 새로운 모델은 **soft alignment**가 적용되어 문맥에 맞는 **유연한 번역을** 실현
- 새로운 모델은 이전의 모델보다 **긴 문장을 해석하는 데 있어 확연히 좋은 성능**을 나타냄

(a): 임의 선택 문장

(b) ~(d): test set에서 뽑은 10~20단어 길이의 문장(unknown words 제거)



Contributions

- Computer Vision 분야에서 주로 사용되던 Attention을 자연어 처리에 도입
- RNNsearch모델: 기존의 RNN모델의 문제점(Bottleneck현상) 해결
  → 기존 모델과 달리 **긴 문장을 번역해도 performance 우수**
- 향후 Transformer(Self-Attention)를 이용한 신경망 기계번역의 아이디어 제공