Listen, Attend and Spell (LAS)

발표자 : 하상천

2020.12.29 (TUE)

Summary

0. 전통적으로 음성 인식 모델은 음향 모델(acoustic model), 발음 모델(pronunciation model), 언어 모델(language model) 등 다양한 구성 요소로 이루어져 있었고 각각의 모델을 따로 학습하여 사용했다.

1. 또한 CTC가 네트워크 출력(word sequence) 을 조건부 독립(conditional independence)으로 가정하고 있어 본질적인 문제가 있다고 지적한다.

2. 본 논문은 음성 인식 시스템을 End-to-End 방식으로 구축하고 네트워크 출력(word sequence) 의 조건부 독립 (conditional independence) 가정을 제거한 아키텍처를 고안한다.

3. 본 논문 이후 Speech 분야는 CTC와 LAS로 나뉜다.

Introduction

- LAS는 sequence to sequence with attention을 기반으로 하고 있다.

- Listener (encoder)는 low level speech signals를 high level features로 바꾸어 주고, Speller (decoder)는 high level features를 attention 기법을 사용하여 확률 분포를 가지고 있는 output utterances로 바꾸어 준다. 그리고 listener와 speller는 함께 학습된다.

- Listener에서는 pyramidal RNN model을 사용하여 time step수를 줄인다.

- 모델 출력이 문자 단위가 되도록 하여 Out-Of-Vocabulary (OOV)도 자동적으로 처리할 수 있다. 또한 조건부 독립 가정을 하는 CTC와 다르게 multiple spelling variants를 생성할 수 있다.
(Ex. "triple a" 에 대해서 모델이 top beams 안에 "triple a" 와 "aaa" 를 생성)

본 논문에서 attention mechanism을 사용하지 않았을 때, 300만개의 발화 training set을 사용하였음에도 불구하고 overfitting이 일어났다고 말하고 있다.
또한 listener쪽에서 pyramid structure를 사용하지 않았을 때, 학습이 너무 느리게 진행되었다고 말하고 있다.

Model

- **Listen**

Input signal x를 high level feature h로 변환해준다.

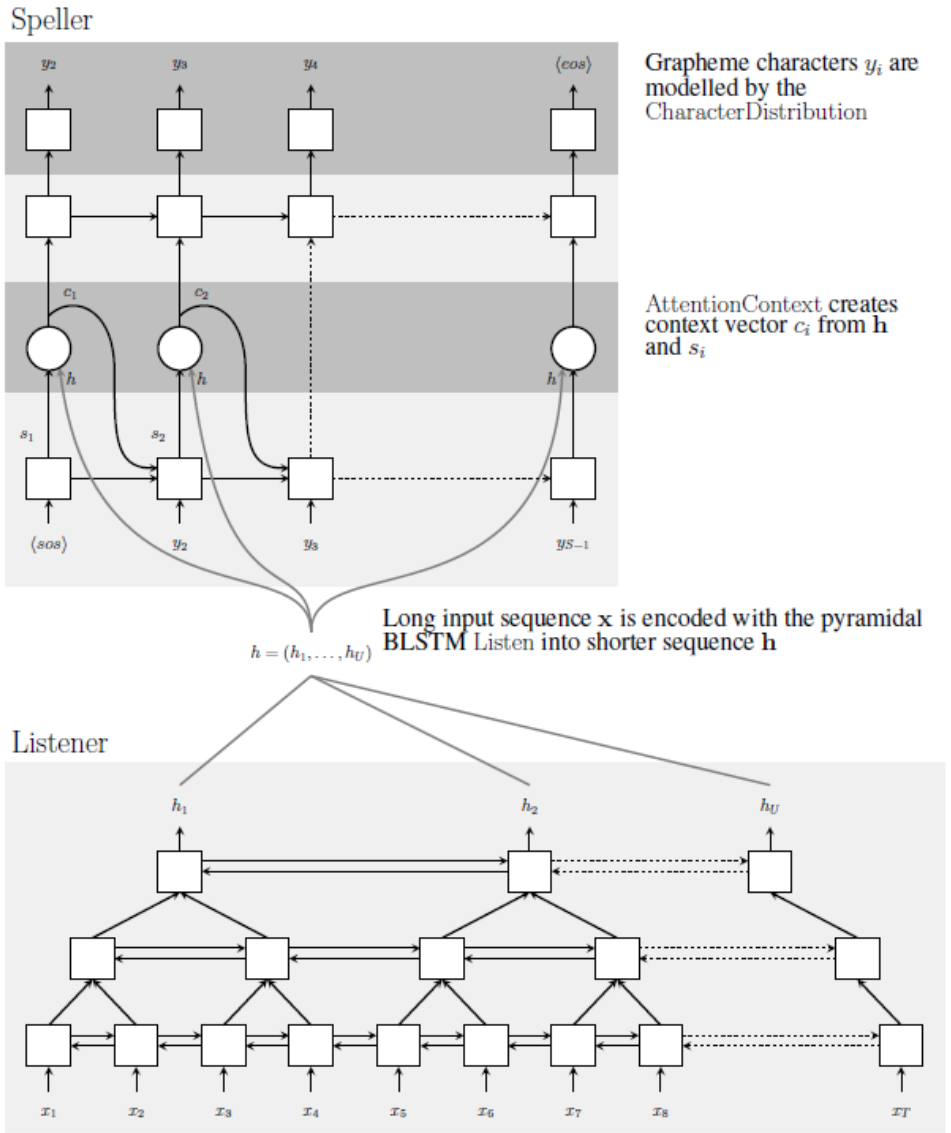
pBLSTM model을 사용하여 input의 길이 T를 U로 줄였다.

본 논문의 모델에서는 3pBLSTM을 쌓아서 총 시퀀스의 길이를 1/8로 줄였다.

또한 이러한 pyramid structure은 계산 복잡성도 줄일 수 있다.

$$\mathbf{h} = \text{Listen}(\mathbf{x})$$

$$h_i^j = \text{pBLSTM}(h_{i-1}^j, \left[h_{2i}^{j-1}, h_{2i+1}^{j-1}\right])$$



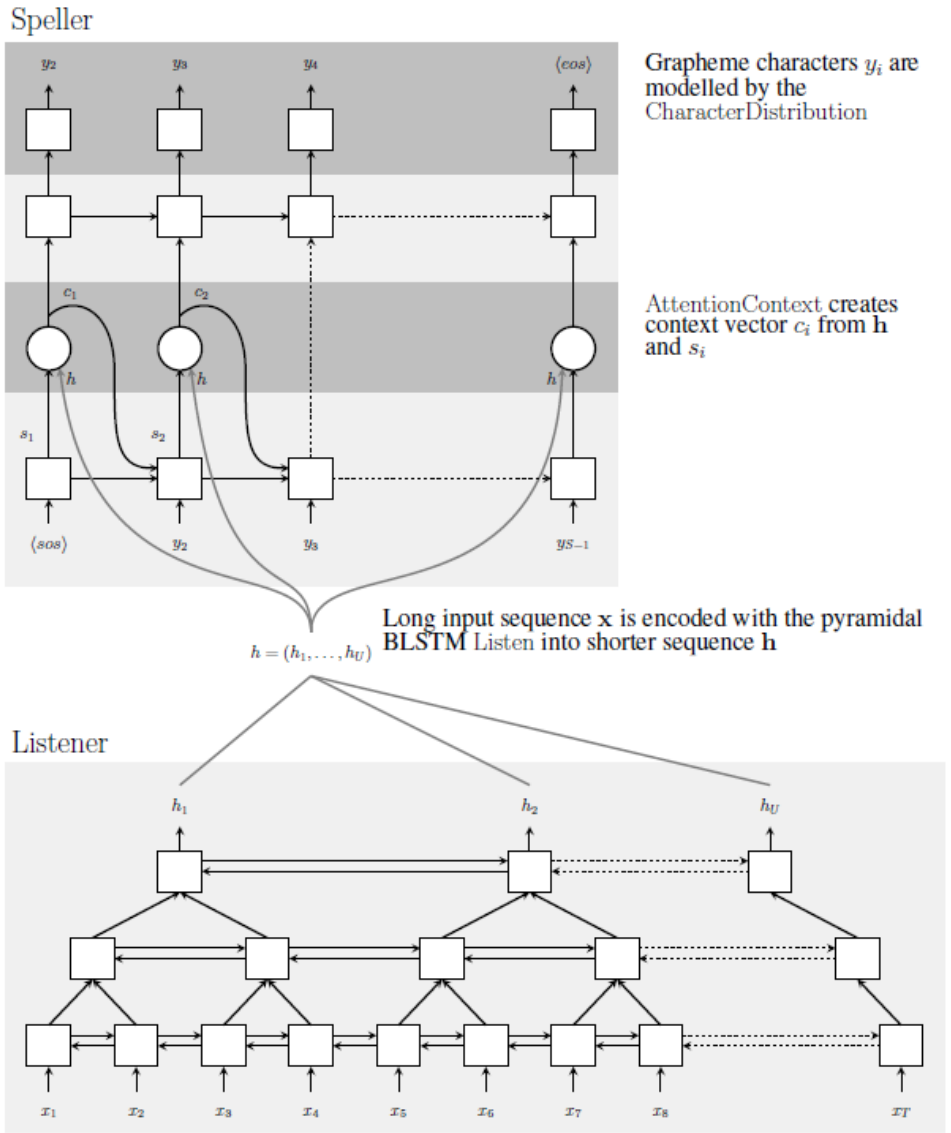
Model

- **Attend and Spell**

각각의 time step마다 이전 단어를 종합하여 다음 단어에 대한 확률 분포를 만든다.

Listener가 변형한 high level feature를 attention을 사용하여 문자로 출력한다.

$$P(\mathbf{y}|\mathbf{x}) = \text{AttendAndSpell}(\mathbf{h}, \mathbf{y})$$



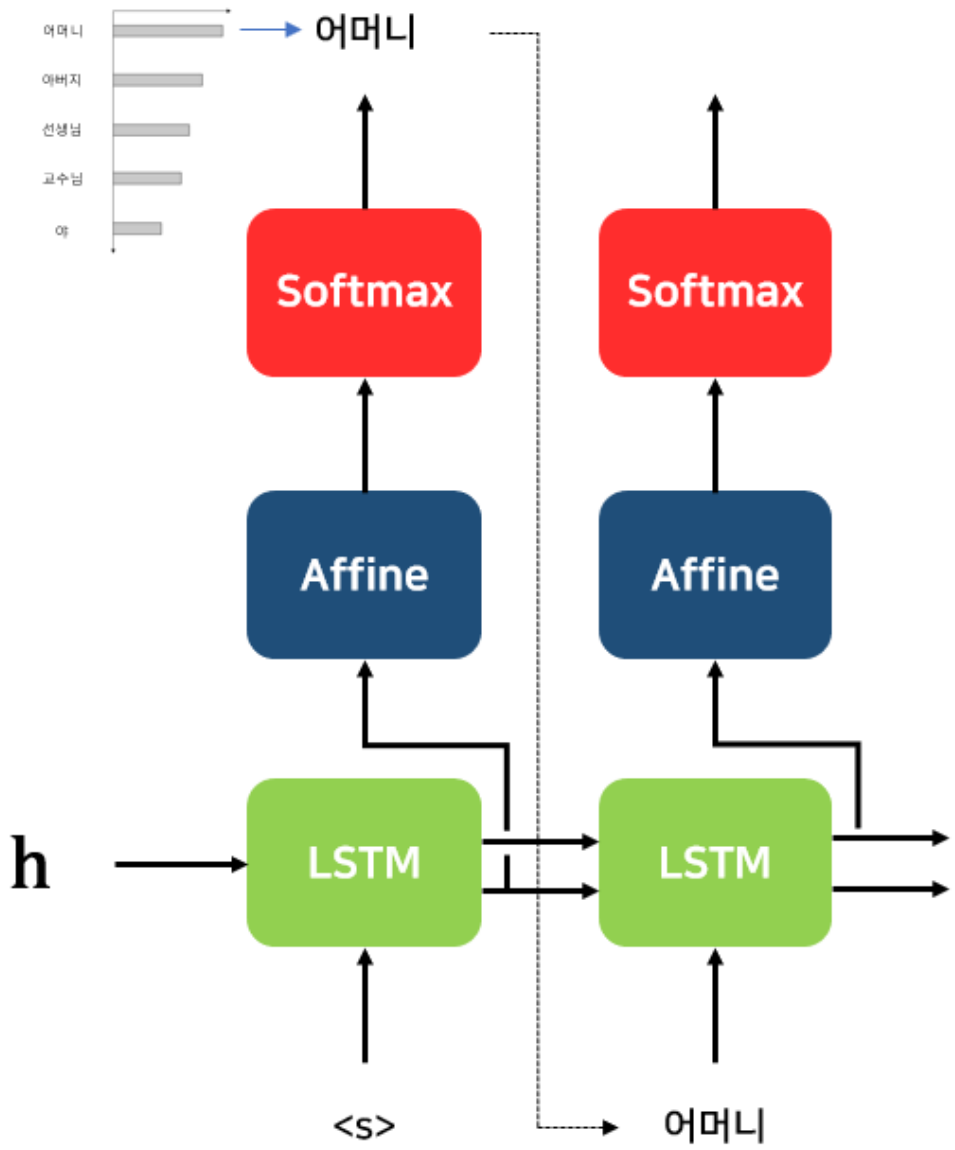
Decoding and Rescoring

- **Beam search**

기본적인 Seq2Seq 모델에서는 가장 확률이 높은 한 가지만 선택한다.
시간복잡도 면에서는 최고지만, 최종 정확도가 떨어질 수 있다.
이러한 방법을 Greedy Decoding이라고 한다.

Beam search는 1등만 선택하는 것이 아니라 나머지에게도
기회를 주는 방법이다.

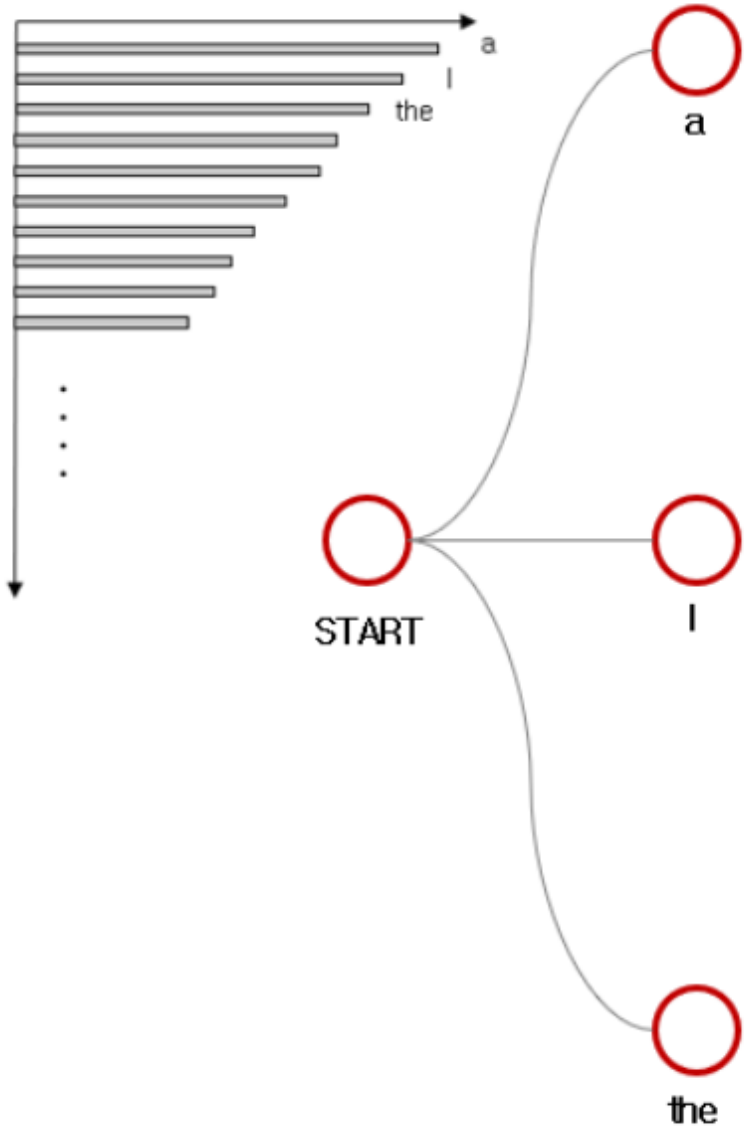
따라서 beam search는 greedy decoding과 모든 경우의 수를
고려하는 방법의 타협점이라고 할 수 있다.



Decoding and Rescoring

- **Beam search**

예를 들어, Beam size K를 3 이라고 하면
예측 값의 확률 분포 중 가장 높은 확률 3개를 고른다.

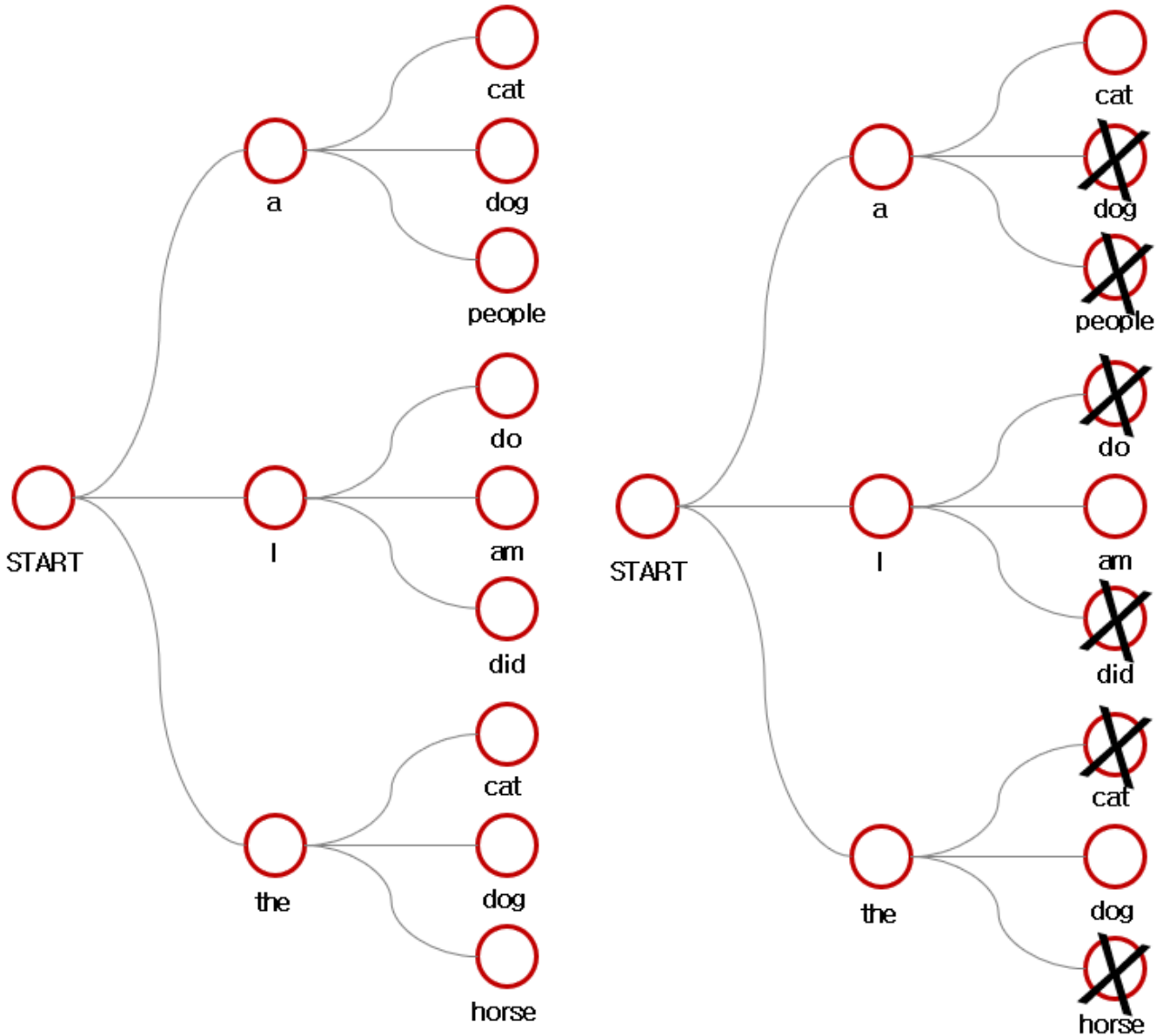


Decoding and Rescoring

- **Beam search**

누적 확률 순으로 상위 3개를 뽑는다.
노드들이 서로 같은 확률을 가지더라도,
어떤 빔을 통해 왔는지에 따라 누적확률은 달라진다.

<eos>를 만난 빔이 3개가 될 때까지 진행하고
<eos>를 만난 빔이 3개가 된다면
총 3개의 후보 중에서 가장 높은 누적 확률을 가진
빔을 최종적으로 선택한다.



Decoding and Rescoring

- **Beam search**

누적확률을 구할 때 확률의 범위는 0.0~1.0 이기 때문에 곱할수록 값이 점점 작아진다.
이러한 길이에 따른 불공평함을 해소하기 위해 Length Penalty라는 개념이 나왔다.

$$lp(Y) = \frac{(5 + |Y|)^{\alpha}}{(5 + 1)^{\alpha}}$$

보통 알파는 1.2정도의 값을 사용하고 5라고 입력 되어 있는 값도 minimum length로 하이퍼파라미터다.

이전의 구했던 누적확률을 length penalty 결과값으로 나누어 주면 된다.

앞서 말씀드린 Greedy Decoding도 K=1인 beam search와 같은 것이다.

Decoding and Rescoring

- **Rescoring**

Beam search를 할 때 dictionary을 사용해서 탐색 공간을 줄일 수 있지만.
본 논문의 실험결과 어느 정도 학습한 후에는 dictionary 없이도 현실의 단어를 잘 생성한다고 말하고 있다.

Beam size 만큼의 후보를 뽑은 후에.
기존의 점수와 language model을 이용해서 구한 점수를 더해
새로운 점수를 매긴다.
가장 높은 점수를 받은 빔을 최종 선택하는 것이다.
여기서 λ는 language model의 가중치로 validation set을 가지고 결정한다.

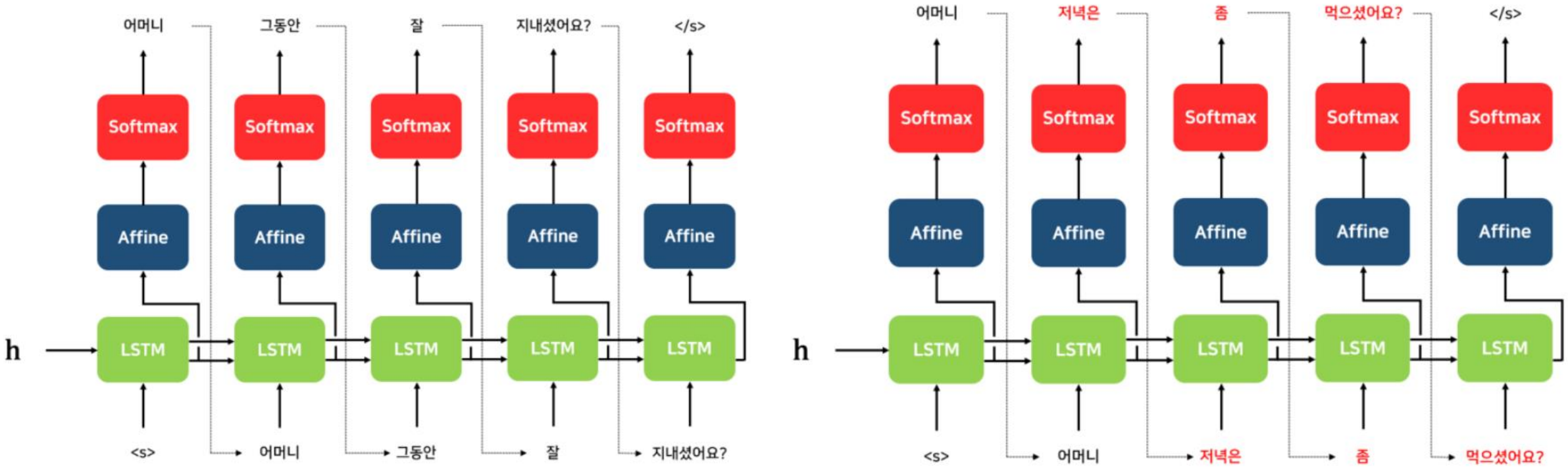
$$s(\mathbf{y}|\mathbf{x}) = \frac{\log P(\mathbf{y}|\mathbf{x})}{|\mathbf{y}|_c} + \lambda \log P_{\mathrm{LM}}(\mathbf{y})$$

Learning

- **Teacher forcing**

티쳐포싱은 target word(ground truth)를 디코더의 다음 입력으로 넣어주는 기법이다.

잘못된 예측이 일어났다면 그 다음 스텝에서도 잘못된 예측이 일어난다.



Learning

- **Teacher forcing**

장점 : 학습이 빠르다
Teacher forcing을 사용하지 않으면 잘못된 예측 값을 토대로 학습하기 때문에 모델의 학습 속도가 느리다.

단점 : 노출 편향 문제 (Exposure Bias Problem)
추론 과정에서는 ground truth를 제공하지 않기 때문에 학습과 추론 단계간의 차이가 발생한다.
따라서 모델의 성능과 안정성을 떨어뜨릴 수 있다.

(2019년 5월에 나온 「Exposure Bias for Neural Language Generation」논문에서는 이런 노출 편향 문제가 생각만큼 큰 영향을 미치지 않는다는 연구 결과를 냈다고 한다)

이러한 기법은 매우 효율적이지만 문자들 간의 관계가 중요한 문장에서 낮은 에러를 예측하기 때문에 학습이 힘들어질 수 있다.
이 문제를 해결하기 위해서 일정 확률로, 실제 라벨 대신에 모델에서 샘플링한 라벨을 입력 값으로 사용한다.(Scheduled sampling)

Experiments

Data	
Train	2,000h
Valid	10h
test	16h

Feature	
Feature size	40
Feature extraction	Log-mel filter bank
Frame length	10ms

HyperParameter	
Encoder layer size	3
Decoder layer size	2
Hidden size	256
Batch size	32
Teacher forcing	1.0 , 0.9

가중치는 -0.1~0.1의 uniform 분포로 초기화
optimizer로 ASGD(Asynchronous Stochastic Gradient Descent), beam size는 32로 진행

Performance

Model	Clean WER	Noisy WER
CLDNN-HMM [20]	8.0	8.9
LAS	16.2	19.0
LAS + LM Rescoring	12.6	14.7
LAS + Sampling	14.1	16.5
LAS + Sampling + LM Rescoring	10.3	12.0

Teacher forcing을 90% 적용한 모델이 100% 적용한 모델보다 성능이 좋은 것을 확인 할 수 있다.

Language model과 90%의 teacher forcing을 적용한 모델이 가장 좋은 퍼포먼스를 보였다.
clean 데이터에서 WER(word error rate) 10.3%, noisy 데이터에서 WER 12.0%를 받았다.

본 논문이 쓰여질 당시 state-of-the-art 모델인 CLDNN-HMM모델과 비교했을 때도 2.3% 밖에 차이가 나지 않았다.
또한, End-to-end 학습이 가능한 모델이라는 점에서 의의가 있다.

본 논문에서는 convolution filter를 사용하면 non-convolutional architecture와 비교했을 때 clean speech에서는 5% WER
noisy speech에서는 7% WER만큼 향상 할 것이라 말하고 있다.

Reference

- Listen, Attend and Spell
https://arxiv.org/abs/1508.01211

- DNN-HMM
https://dongchans.github.io/2019/115/

- Beam Search
https://blog.naver.com/sooftware/221809101199

- Teacher Forcing
https://blog.naver.com/sooftware/221790750668