



## 5-3 Course Presentation

### Basic models

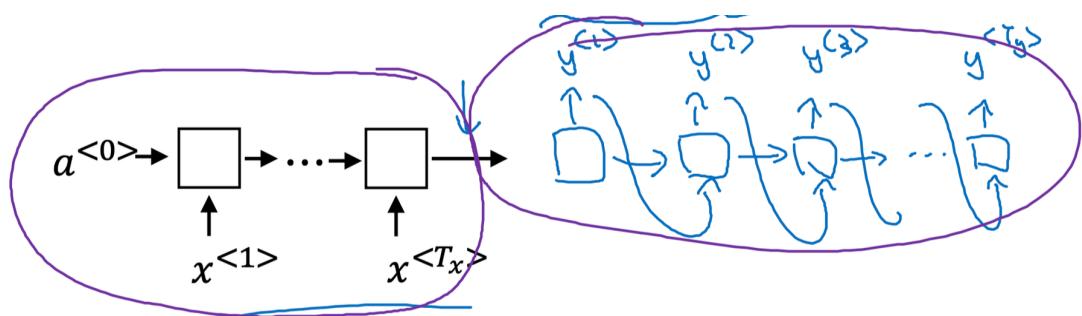
#### Sequence to sequence model

$x^{<1>} \quad x^{<2>} \quad x^{<3>} \quad x^{<4>} \quad x^{<5>}$   
Jane visite l'Afrique en septembre

→ Jane is visiting Africa in September.

$y^{<1>} \quad y^{<2>} \quad y^{<3>} \quad y^{<4>} \quad y^{<5>} \quad y^{<6>}$

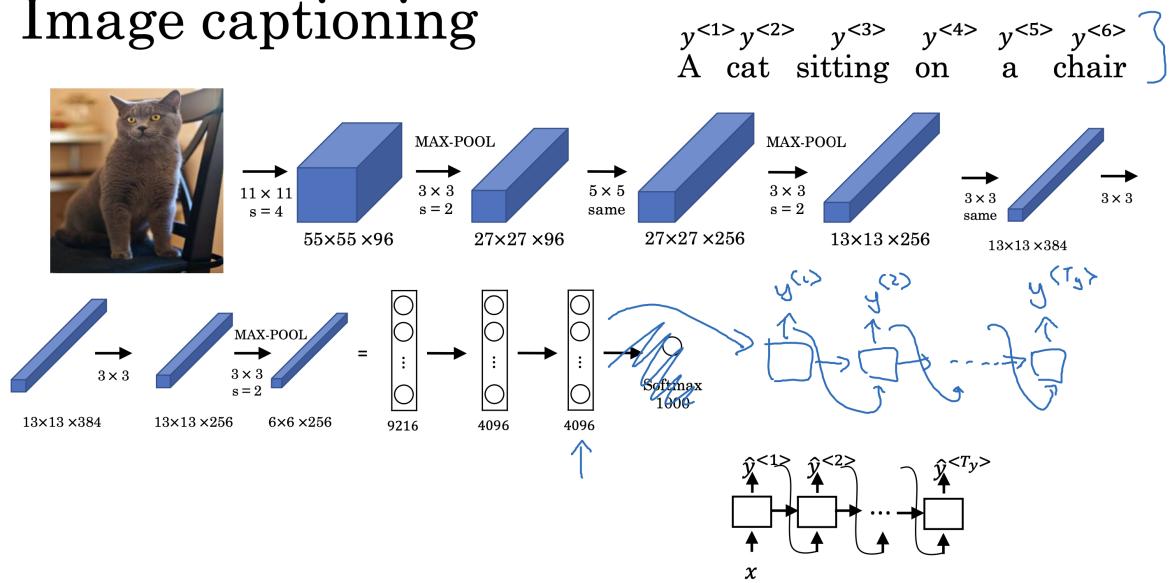
- ‘Jane visite l’Afrique en septembre’라는 프랑스어로 된 문장을 영어 문자으로 변환하고 싶다면, 프랑스어로 된 문장 시퀀스를  $x^{<1>}$ 부터  $x^{<5>}$ 까지 표시하고,  $y^{<1>}$ 에서  $y^{<6>}$ 까지 사용해서 output 시퀀스 단어 표시
- 그러면 어떻게 새로운 network를 학습해서 시퀀스  $x$ 를 입력으로 하고 시퀀스  $y$ 를 출력 할 수 있을까?



- 이 모델은 인코더(Encoder) 네트워크와 디코더(Decoder) 네트워크로 구성
- 인코더를 통해서 프랑스 단어를 입력받고, 디코더를 통해서 영어 단어 하나하나를 출력

- 해당 모델은 충분한 프랑스어 문장과 영어 문장의 데이터셋이 있을 때, 효과적!

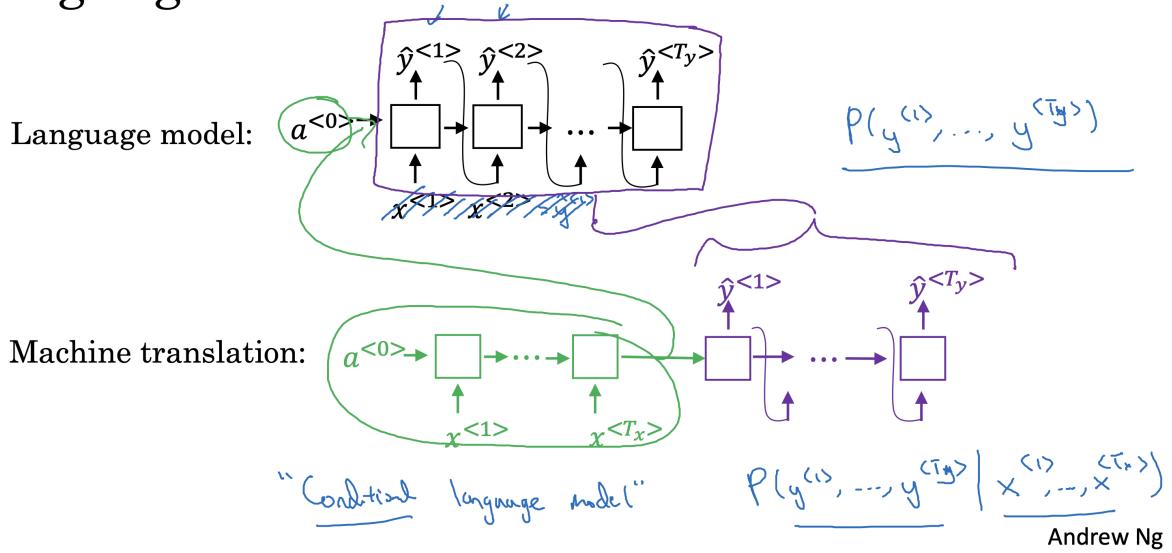
## Image captioning



- sequence-to-sequence 모델과 비슷한 구조로 Image captioning(이미지 캡션)도 수행 가능
- 슬라이드 내용 설명:
  - pre-trained된 AlexNet을 사용해서, 마지막 softmax layer를 삭제한 후에 AlexNet으로 4096 차원의 특성 벡터를 출력하게 되는데, 이 부분이 이미지의 인코더에 해당
  - RNN 모델을 네트워크로 사용해서 4096차원의 특성벡터를 입력으로 사용해서 (AlexNet으로 얻어짐) 이미지를 설명하는 단어들이 출력으로 나오게 한다

## Picking the most likely sentence

# Machine translation as building a conditional language model



- 기계번역(Machine translation)은 조건부 언어모델(conditional language model)로 생각할 수 있는데, 기계번역 모델의 구성을 보면 뒤쪽의 디코더 네트워크가 language 모델과 유사한 것을 알 수 있다
- 구조 파악:
  - language model은 0벡터에서 시작
  - machine translation은 인코더 네트워크를 통과한 출력물
  - 이 값들이 디코더 네트워크의 입력

## Finding the most likely translation

Jane visite l'Afrique en septembre.

$$P(y^{<1>} \dots, y^{<T_y>} | x)$$

- Jane is visiting Africa in September.
- Jane is going to be visiting Africa in September.
- In September, Jane will visit Africa.
- Her African friend welcomed Jane in September.

$$\arg \max_{y^{<1>} \dots, y^{<T_y>}} P(y^{<1>} \dots, y^{<T_y>} | x)$$

- 입력으로 프랑스어 문장  $x$ 와 출력인 영어 문장  $y$ 가 주어진다면, 모델은 입력(프랑스어 문장)에 대한 출력의(영어 문장) 확률을 알려준다.

$$P(y^{<1>} , \dots, y^{<Ty>} | x^{<1>} , \dots, x^{<Tx>})$$

- 결국 우리는 위 모델을 통해서 프랑스어 문장을 영어로 번역하려면, 가장 높은 확률의 번역인 아래 값을 찾아야 한다!

$$P(y^{<1>} , \dots, y^{<Ty>} | x)$$

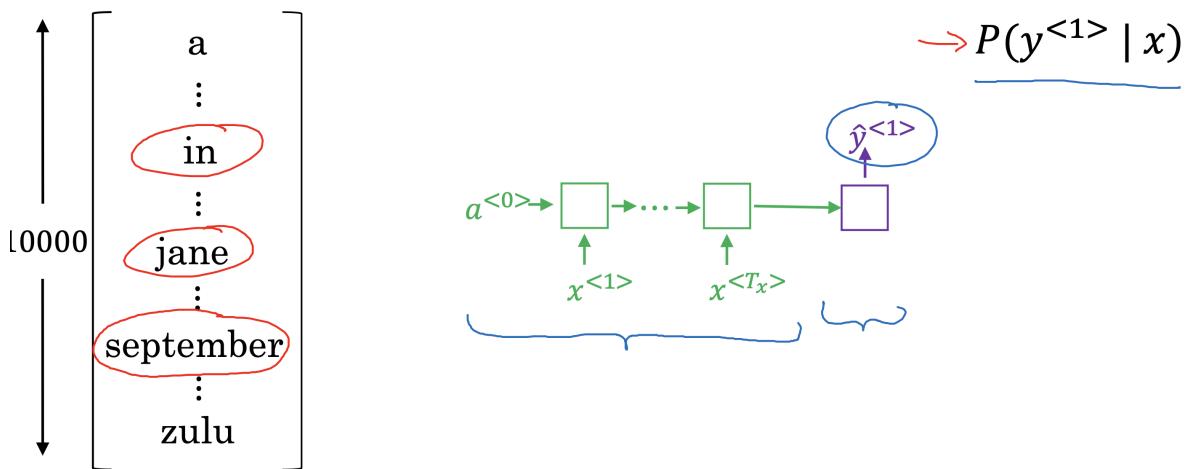
- 정리:
  - 모델은 입력인 프랑스어 문장( $x$ )에 대해서 영어 문장( $y$ )들의 조건부 확률 출력
  - 모델의 확률 분포를 얻어서 샘플링했을 때, 괜찮은 번역을 얻을 수도 있고, 좋지 않은 번역을 얻을 수도 있다(확률이니까)
  - 좋은 번역을 얻기 위해서는 조건부 확률을 최대로하는 영어 문장 찾아야 한다
  - 즉, 기계 번역 시스템을 개발할 때, 해야 할 일 중의 하나는 **조건부 확률을 최대화하는 알고리즘 고안**

→ **Beam Search**

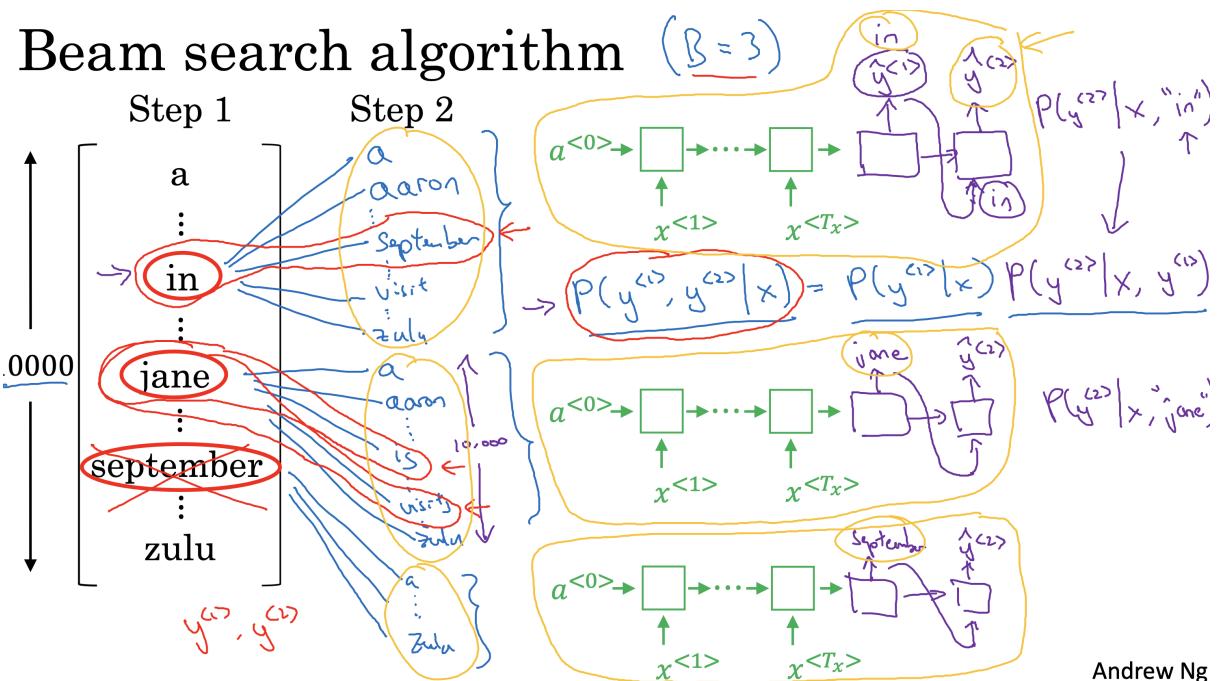
## Beam Search

**Beam search algorithm**       $B = 3$       (beam width)

Step 1



- Beam Search는 가장 확률이 높은 output을 찾기위해서 사용되는 알고리즘(다양한 경우 고려)
- 동작원리:
  - 매개변수  $B = 3$ : Beam width → Beam search가 3개의 가장 높은 가능성을 고려한다는 의미
    - 그래서 Step1 과정에서 가장 확률( $P(y^{<1>}|x)$ )이 높은 단어 3개 'in', 'jane', 'september'를 선택



- Step2 단계에서는 Step1에서 선택한 3개의 단어를 Step2의 입력으로 사용해서 가장 높은 확률(아래)을 가지는 3개의 출력을 다시 선택

$$(P(y^{<1>}, y^{<2>} | x) = (P(y^{<1>} | x)(P(y^{<2>} | x, y^{<1>}))$$

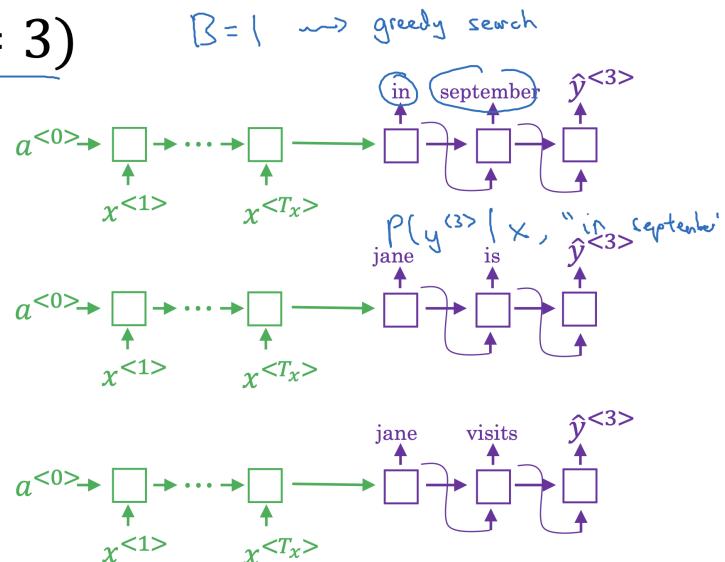
- 결과적으로 'in Steptember', 'jane is', 'jane visits'가 선택되었다고 가정

## Beam search ( $B = 3$ )

in september  
 aaron  
 jane  
 zulu

jane is  
 visits  
 zulu

jane visits  
 africa  
 zulu



$$P(y^{<1>} , y^{<2>} | x)$$

jane visits africa in september. <EOS>

- Step3도 앞에서와 마찬가지로 동일하게 진행하고, 가장 높은 가능성을 가진 3개의 확률을 선택하게 된다.
- 계속 진행하다가 <EOS>를 만나게 되면 종료하고 해당 영어 문장을 출력
  - EOS : End Of Sentence

## Refinements to Beam Search

- Beam search를 더욱 극대화할 수 있는 팁 설명

## Length normalization

$$\arg \max_y \prod_{t=1}^{T_y} P(y^{<t>} | x, y^{<1>} , \dots , y^{<t-1>})$$

$P(y^{<1>} \dots y^{<T_y>} | x)$

*log*

- Beam Search는 결국 조건부 확률의 최대가 되는 경우를 찾는 것이므로  $P(\text{확률})$ 는 모든 값이 1보다 작고, 결국  $P$ 를 곱하다 보면 그 값은 1보다 훨씬 작아지게 된다.
- 특히 단어가 많을수록(경우가 많아지니까 곱셈 연산 많아짐) 값은 기하급수적으로 작아지며 컴퓨터가 값을 정확하게 표현하기에 너무 작아질 수도 있다.
- 

$$\arg \max_y \sum_{t=1}^{T_y} \log P(y^{<t>} | x, y^{<1>}, \dots, y^{<t-1>})$$

$T_y = 1, 2, 3, \dots, 30.$

$$\rightarrow \frac{1}{T_y^\alpha} \sum_{t=1}^{T_y} \log P(y^{<t>} | x, y^{<1>}, \dots, y^{<t-1>})$$

- 위의 문제점을 방지하기 위해서 확률  $\log$ 를 취해서 최대가 되는 값을 찾기!
- $\log$  함수도 증가하는 함수이기 때문에  $\log P(y|x)$ 를 최대화하는 것은  $P(y|x)$ 를 최대화하는 것과 동일하며,  $\log$ 를 취해주면 수치적으로 안정적이게 된다.( $\log$  함수의 그래프를 보면 이해 가능)
- $\log$ 를 취해도 단어가 많아질수록 같은 문제가 발생할 수 있는데 이를 해결하기 위해서 단어의 수인  $T_y$ 로 정규화를 하는 것이다
  - 로그함수를  $T_y$ 로 나눠주기
- B의 크기:
  - 아주 크다 : 많은 가능성을 고려하게 되고, 좋은 결과를 얻지만 느려지고, 메모리 많이 차지
  - 매우 작다 : 고려하는 경우의 수 적기 때문에 좋지 않은 결과, 속도 빠르고 메모리 덜 차지
- 실제 시스템에서는 10개를 사용하기도 하고, 최고의 결과를 위해서 1000~3000개를 사용하기도 함

## Bleu(Bilingual Evaluation Understudy) Score

- 기계번역은 번역값을 여러 개 만들수 있는데, 하나의 정답이 있는 이미지 인식과는 달리 여러가지 좋은 정답이 있다면 어떻게 평가할 수 있을까?
- 이때 사용되는 것이 보편적으로 **Bleu Score**
  - **Bleu Score**는 번역이 얼마나 괜찮은지 측정하는 점수를 측정할 수 있게 해준다
  - 만약 예측 결과가 사람이 제공한 레퍼런스와 가깝다면 **Bleu Score**의 점수는 높다
- Bleu Score 점수 산정 방식 : Modified Precision
  - 각 단어에 대해서 중복을 제거하고, reference 문장에 나타난 최대 횟수만큼 점수를 부여한다.

Reference 1: The cat is on the mat. ↙

Reference 2: There is a cat on the mat. ↙

MT output: the the the the the the.

Precision:

Modified precision:

- Ref1에서는 'the'가 2번 나타나고, Ref2에서는 'the'가 1번 나타난다. 따라서, Modified precision은 2/7가 된다.

## Bleu score on bigrams

Example: Reference 1: The cat is on the mat. ←

Reference 2: There is a cat on the mat. ←

MT output: The cat the cat on the mat. ←

	Count	Count <sub>clip</sub>	
the cat	2 ←	1 ←	
cat the	1 ←	0	
cat on	1 ←	1 ←	
on the	1 ←	1 ←	
the mat	1 ←	1 ←	

$\frac{4}{6}$

- 위에서의 경우는 각 단어를 개별적으로 살펴보아서 단어의 순서를 고려하지 않았는데 Bleu Score로 평가할 때, 단어 쌍으로 Bleu Score를 정의해서 사용할 수 있다.
- bigrams( $n=2$ )에서 Bleu Score는 MT output의 각 단어들을 서로 근접한 두 개의 단어들로 묶어서, Reference에 얼마나 나타나는지 체크해서 Modified Precision을 계산한다.

## Bleu details

$p_n$  = Bleu score on n-grams only

$P_1, P_2, P_3, P_4$

Combined Bleu score:  $BP \exp\left(\frac{1}{4} \sum_{n=1}^4 p_n\right)$

$BP$  = brevity penalty

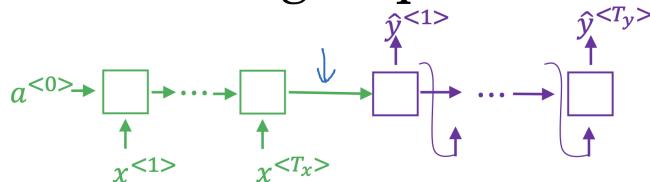
$$BP = \begin{cases} 1 & \text{if } \underline{\text{MT\_output\_length}} > \underline{\text{reference\_output\_length}} \\ \exp(1 - \text{MT\_output\_length}/\text{reference\_output\_length}) & \text{otherwise} \end{cases}$$

- $P_n$ 이 n-grams에서 bleu score이고, 만약  $n=4$ 라면 Bleu Score는 아래와 같이 계산 할 수 있다.
  - BP는 Brevity Penalty로 짧은 번역이 높은 score를 얻는 것에 대해 패널티를 부여하는 것을 의미

$$\text{Combined Bleu score} : BPexp(41n = 1 \sum 4pn)$$

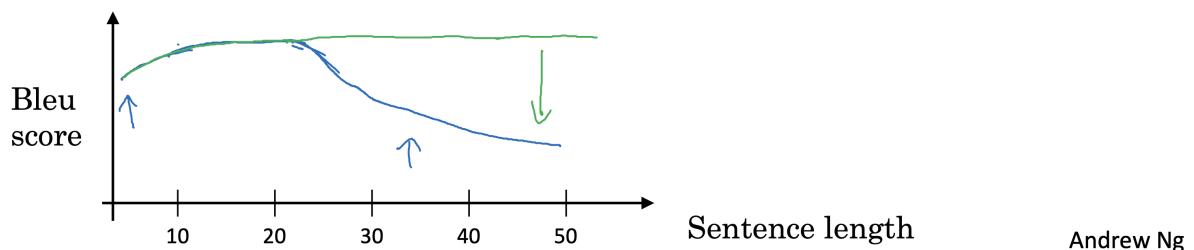
## Attention Model

### The problem of long sequences



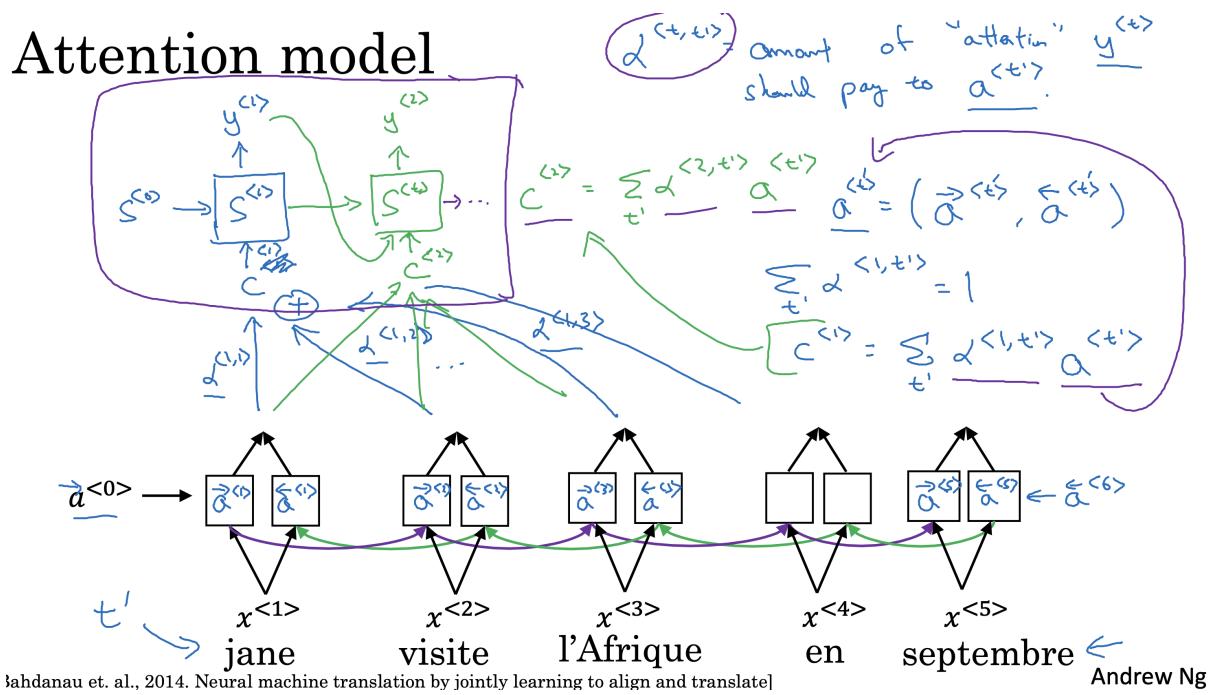
Jane s'est rendue en Afrique en septembre dernier, a apprécié la culture et a rencontré beaucoup de gens merveilleux; elle est revenue en parlant comment son voyage était merveilleux, et elle me tente d'y aller aussi.

Jane went to Africa last September, and enjoyed the culture and met many wonderful people; she came back raving about how wonderful her trip was, and is tempting me to go too.



- 대부분의 기계번역은 Encoder-Decoder 구조를 사용하고, 하나의 RNN에서 입력 문장을 읽고 다른 RNN에서 문장을 추력
  - 해당 아키텍처는 일반적으로 짧은 문장에서 잘 작동하고, 입력 문장의 길이가 길어질수록 성능이 낮아져서 Bleu Score가 낮아짐
- Attention model을 사용하면 긴 문장에서도 성능 유지 가능
  - 사람이 번역할 때 문장 중간중간을 번역하는 것처럼 Attention model도 사람과 유사하게 번역
  - 매 예측시점마다 인코더에서의 입력 문장을 다시 참고하는데, 이때 전체 입력 문장을 참고하는 것이 아닌 예측할 단어와 연관되는 부분만 집중(Attention)해서 참고

# Attention model

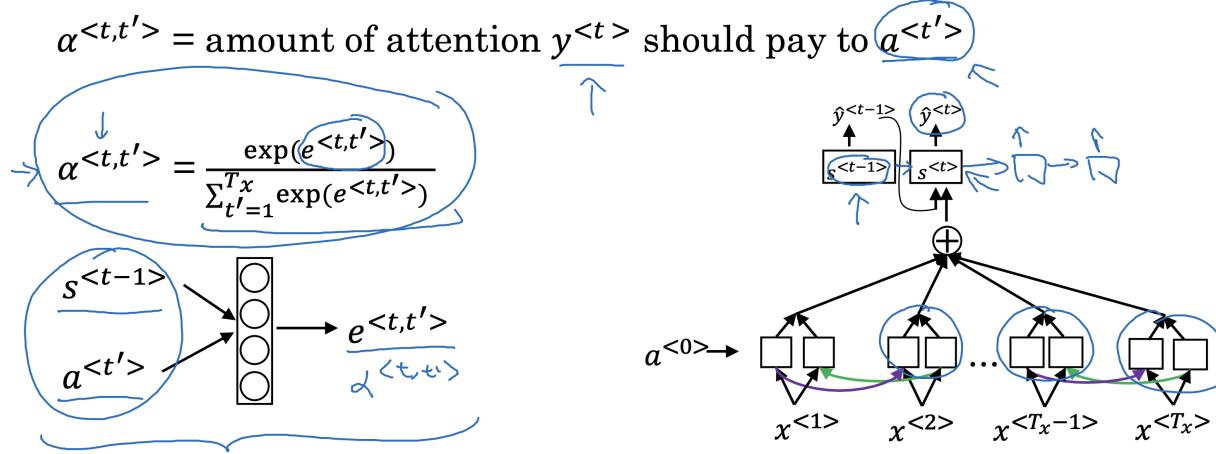


- Encoder Network 부터 확인해보면 Forward / Backward에서 계산되는 activation은 각각  $a^{\rightarrow}, a^{\leftarrow}$ 로 표시하고 이를 합쳐서  $a^{(t)} = (a^{\rightarrow(t)}, a^{\leftarrow(t)})$ 로 표시( $t$ 는 input  $x$ 의 time step을 의미)
- Decoder의 입력으로 사용되는  $C$ 는 아래와 같이 계산

$$c^{(1)} = \sum_{t'} \alpha^{(1, t')} a^{(t')}$$

- $\alpha$ 는 Attention parameter로써  $C$ 가 activation과 feature에 얼마나 의존적인지 나타내며,  $y^{(t)}$  예측에 얼마나 attention 해야 할지를 나타내는 정도(amount)이다.

## Computing attention $\underline{\alpha^{<t,t'>}}$



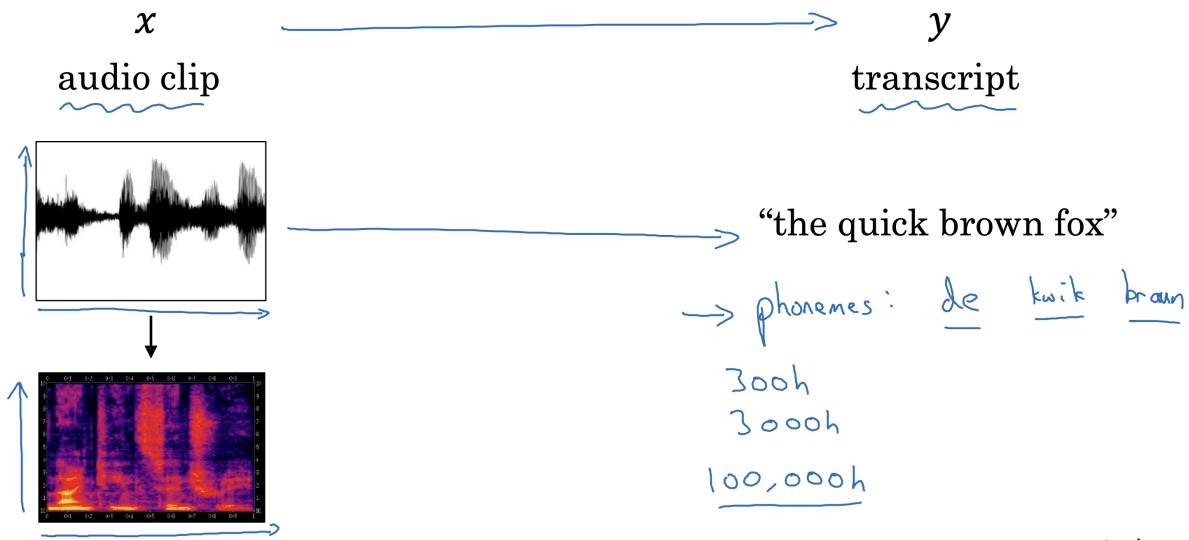
- 계산방법

$$\alpha^{<t,t'>} = \frac{\sum_{t'=1}^{T_x} \exp(e^{<t,t'>})}{\exp(e^{<t,t'>})}$$

- $\alpha^{<t,t'>}$ 는 softmax 확률로 계산되어 총합이 1이다.
- $e^{<t,t'>}$ 은 작은 신경망을 통해서 구할 수 있으며, 이전 layer의 hidden activation인  $s^{<t-1>}$ 과  $a^{<t'>}$ 을 입력으로 함

## Speech recognition

# Speech recognition problem



Andrew Ng

- 음성인식은 오디오 클립  $x$ 를 통해서 transcript를 찾는 것이다
- 일반적으로 입력 데이터를 주파수 별로 분리하는 것
  - 오디오 클립 아래 그래프:
    - x축 : 시간
    - y축: 주파수
- 구현을 어떻게 할 수 있을까?(설명 빈약..)
  1. Attention model
  2. CTC cost

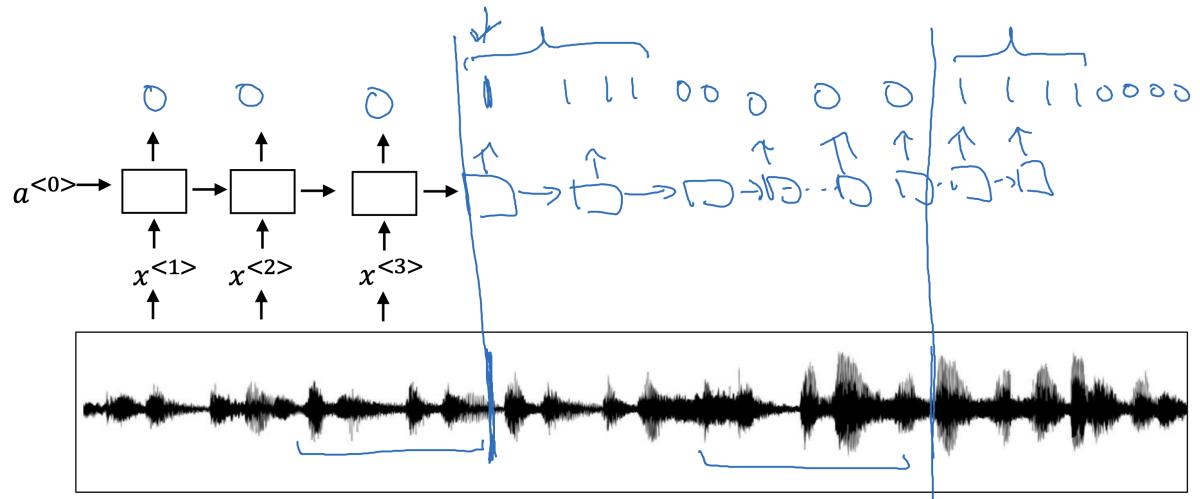
## Trigger word detection

# What is trigger word detection?



- Trigger word detection : 기계를 깨울 수 있는 방법 의미
  - 군대에서 Gini 같은 거..!

## Trigger word detection algorithm



- RNN 모델을 보면 오디오 클립  $x$ 가 입력으로 사용
- label인  $y$ 를 정의하는데, trigger word를 누군가가 말할 때 trigger word를 말하기 전의 시점은 모두 label 값을 0으로 설정하고, 말하고 난 직후 시점을 label 값을 1로 설정
- 실제로 잘 동작하지 않는데, label이 불균형하게 분포하고 있기 때문에 1에 비해서 0이 훨씬 더 많다.