Show Attend and Tell

첨언 자료

Enjoy your stylish business and campus life with BIZCAM

Image caption

Image caption

Show, Attend and Tell: Neural Image Caption Generation with ...

Computer Science > Machine Learning. **arXiv**:1502.03044 (cs). [Submitted on 10 Feb 2015 (v1), last revised 19 Apr 2016 (this version, v3)] ...

K Xu 저술 · 2015 · 5924회 인용 · 관련 학술자료

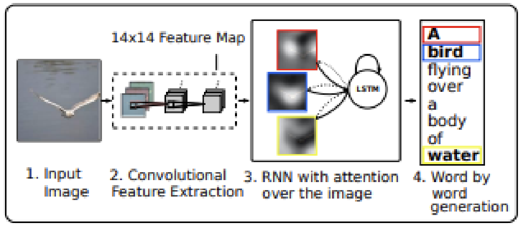
Show, Attend and Tell: Neural Image Caption Generation with Visual Attention

Kelvin Xu	KELVIN.XU@UMONTREAL.CA
Jimmy Lei Ba	JIMMY@PSI.UTORONTO.CA
Ryan Kiros	RKIROS@CS.TORONTO.EDU
Kyunghyun Cho	KYUNGHYUN.CHO@UMONTREAL.CA
Aaron Courville	AARON.COURVILLE@UMONTREAL.CA
Ruslan Salakhutdinov	RSALAKHU@CS.TORONTO.EDU
Richard S. Zemel	ZEMEL@CS.TORONTO.EDU
Yoshua Bengio	FIND-ME@THE.WEB

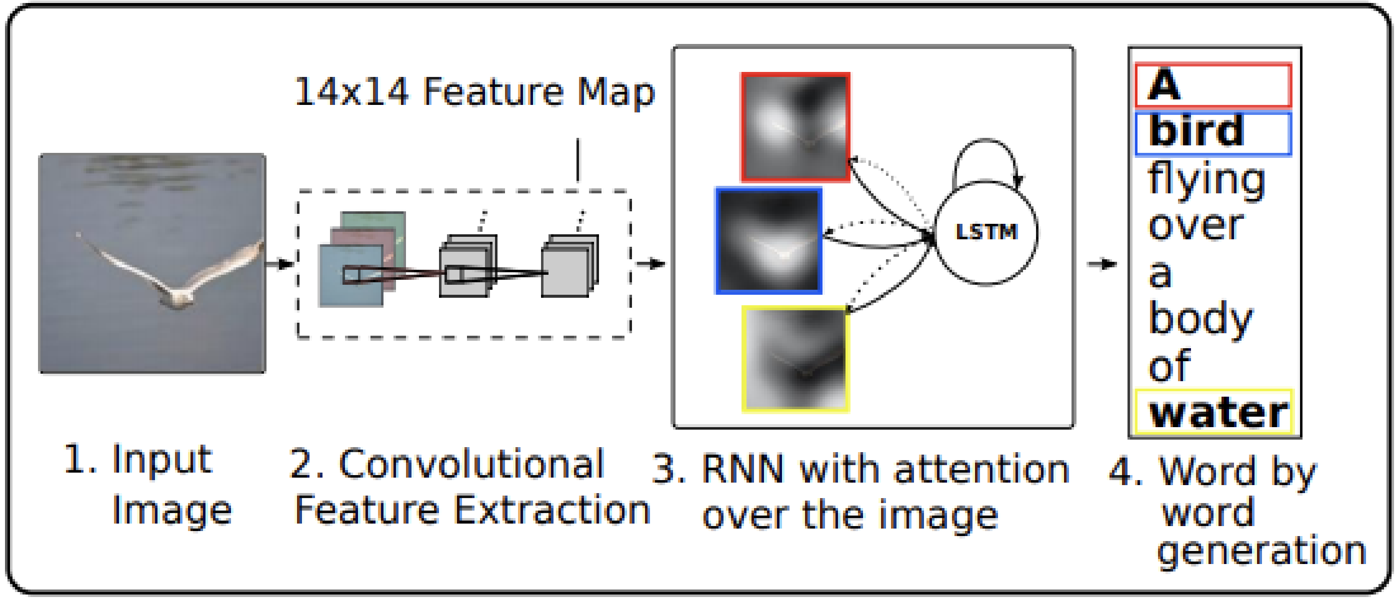
Abstract

Inspired by recent work in machine translation and object detection, we introduce an attention based model that automatically learns to describe the content of images. We describe how we can train this model in a deterministic manner using standard backpropagation techniques and stochastically by maximizing a variational lower bound. We also show through visualization how the model is able to automatically learn to fix its gaze on salient objects while generating the corresponding words in the output sequence. We

Figure 1. Our model learns a words/image alignment. The visualized attentional maps (3) are explained in section 3.1 & 5.4



Outline

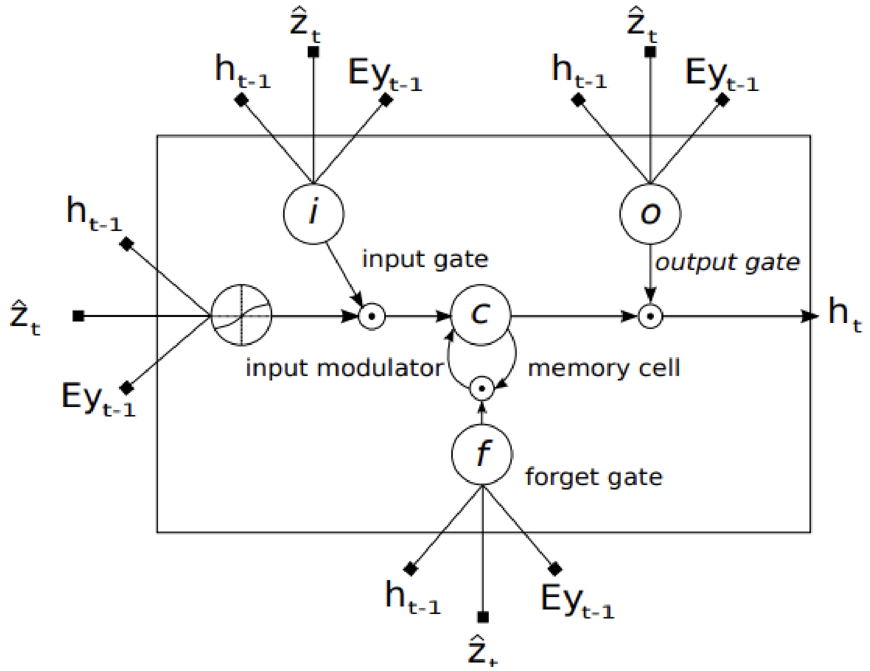

14x14 Feature Map
LSTM
A
bird
flying
over
a
body
of
water
1. Input Image
2. Convolutional Feature Extraction
3. RNN with attention over the image
4. Word by word generation

Encoder

- Encoder CNN 은 주어진 이미지를 input 으로 받아 output 으로 feature vector a 를 내보낸다.
- layer 은 총 L 개 , 각 filter 마다 D 개의 뉴런
- $a = \{a_1, \dots \dots . a_L \}, a_i \in R^D$

Decoder

- Decoder 로 LSTM 을 사용한다.
- 매 time stamp t 마다 caption vector y 의 한 element 인 y_t를 생성한다.



Decoder

- i_t : *input*
- f_t : *forget*
- c_t : *memory*
- o_t : *output*
- h_t : *hidden state*
- $\hat{z} \in R^D$: *context vector*
- $E \in R^{m*k}$: embedding matrix
- m : *embedding dimention*
- n : *LSTM dimensionality*
- σ : *logistic sigmoid activatoin*
- $\odot$: *element wise multiplication*
- $T_{s,t} : R^s \to R^t$: *affine transformation* ($T_{n,m}(x) = Wx + b$)

$$\begin{pmatrix} \mathbf{i}_t \\ \mathbf{f}_t \\ \mathbf{o}_t \\ \mathbf{g}_t \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} T_{D+m+n,n} \begin{pmatrix} \mathbf{E}\mathbf{y}_{t-1} \\ \mathbf{h}_{t-1} \\ \hat{\mathbf{z}}_t \end{pmatrix}$$

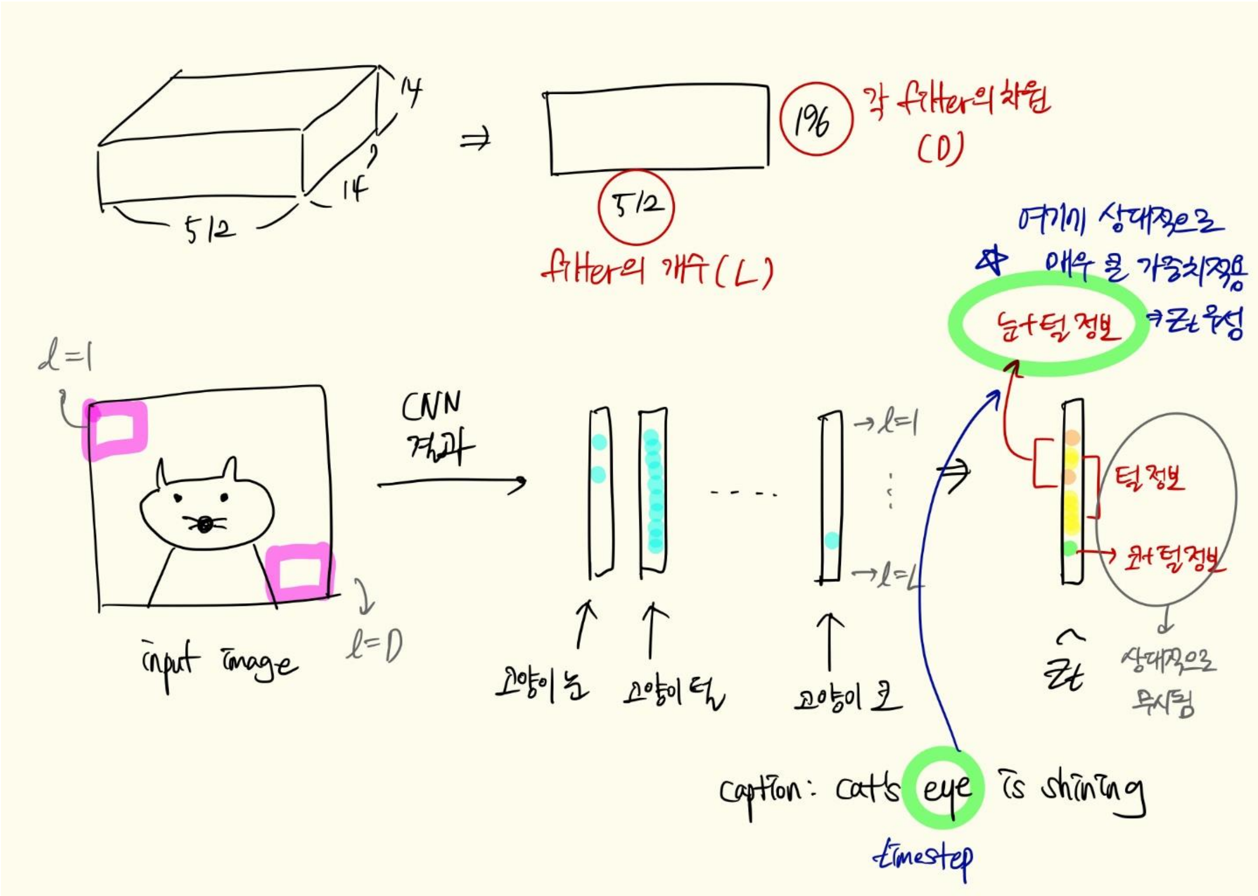
$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t).$$

Decoder

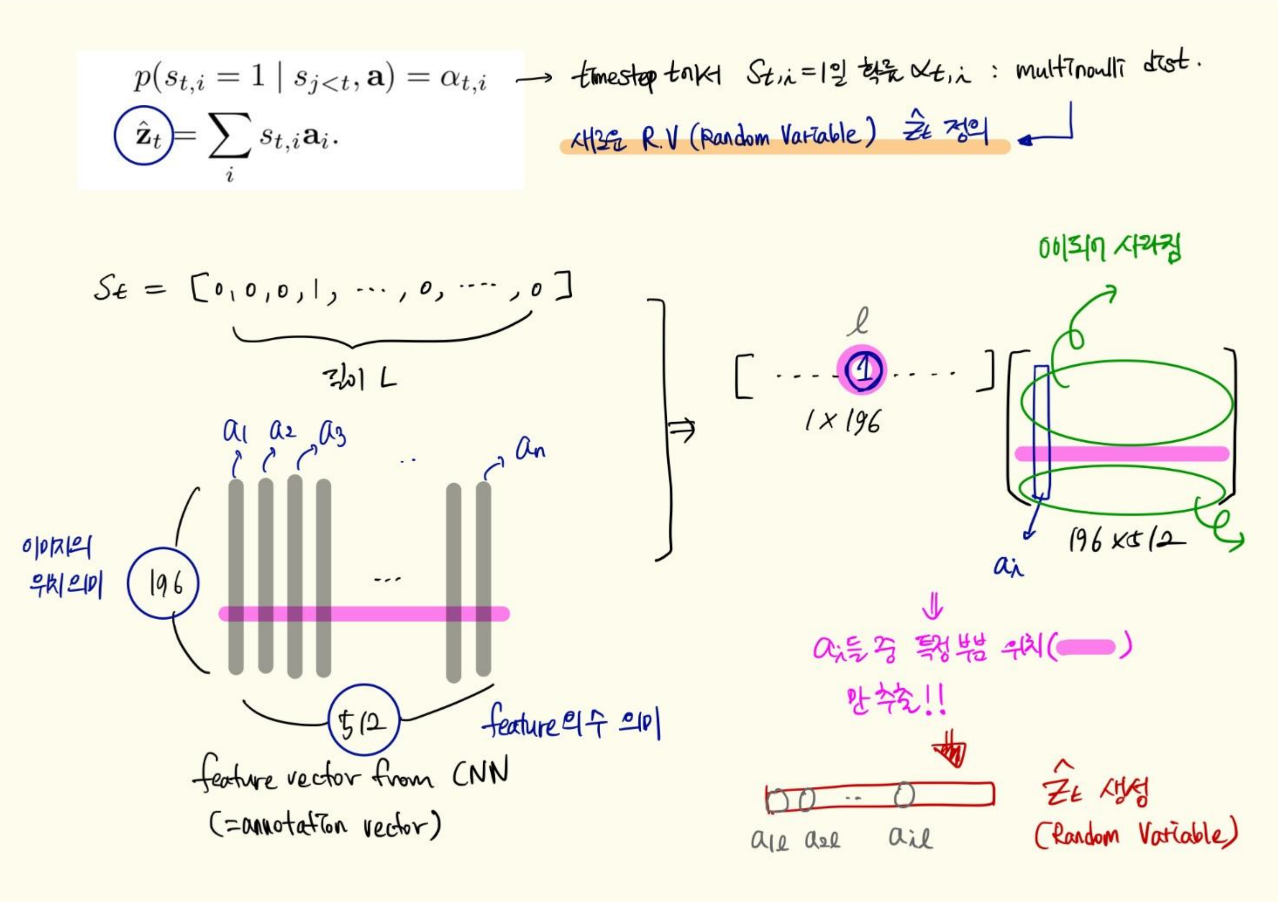
- $\hat{z}_t = \emptyset(a, \alpha_t), where\ \alpha_{ti} = \frac{\exp(f_{att}(a_i, h_{t-1}))}{\sum_k \exp(f_{att}(a_k, h_{t-1}))}$
- α_t : a 의 weight 벡터. 어디로 attend 할지를 결정하는 값
- f_{att} : a 와 h_{t-1}을 사용해 weight vector α 를 계산하기 위한 attention model
- $\emptyset$: α 와 α_t 를 받아 $\hat{z}$ 를 계산하는 메커니즘(모델)

Outline



https://ahjeong.tistory.com/8

Hard attention model



https://ahjeong.tistory.com/8

Hard attention model

$$p(s_{t,i} = 1 \mid s_{j<t}, \mathbf{a}) = \alpha_{t,i}$$

$$\hat{\mathbf{z}}_t = \sum s_{t,i}\mathbf{a}_i.$$

Hard attention model

$$
\begin{aligned}
L_s &= \sum_s p(s \mid \mathbf{a}) \log p(\mathbf{y} \mid s, \mathbf{a}) \\
&\leq \log \sum_s p(s \mid \mathbf{a}) p(\mathbf{y} \mid s, \mathbf{a}) \\
&= \log p(\mathbf{y} \mid \mathbf{a})
\end{aligned}
$$

$$
\frac{\partial L_s}{\partial W} = \sum_s p(s \mid \mathbf{a}) \left[\frac{\partial \log p(\mathbf{y} \mid s, \mathbf{a})}{\partial W} + \log p(\mathbf{y} \mid s, \mathbf{a}) \frac{\partial \log p(s \mid \mathbf{a})}{\partial W} \right].
$$

Hard attention model

$$\tilde{s}_t \sim \text{Multinoulli}_L(\{\alpha_i\})$$

$$\frac{\partial L_s}{\partial W} \approx \frac{1}{N} \sum_{n=1}^{N} \left[\frac{\partial \log p(\mathbf{y} \mid \tilde{s}^n, \mathbf{a})}{\partial W} + \log p(\mathbf{y} \mid \tilde{s}^n, \mathbf{a}) \frac{\partial \log p(\tilde{s}^n \mid \mathbf{a})}{\partial W} \right]$$

Hard attention model

$$b_k = 0.9 \times b_{k-1} + 0.1 \times \log p(\mathbf{y} \mid \tilde{s}_k, \mathbf{a})$$

$$\frac{\partial L_s}{\partial W} \approx \frac{1}{N} \sum_{n=1}^{N} \left[\frac{\partial \log p(\mathbf{y} \mid \tilde{s}^n, \mathbf{a})}{\partial W} + \lambda_r (\log p(\mathbf{y} \mid \tilde{s}^n, \mathbf{a}) - b) \frac{\partial \log p(\tilde{s}^n \mid \mathbf{a})}{\partial W} + \lambda_e \frac{\partial H[\tilde{s}^n]}{\partial W} \right]$$

Soft attention model

- Soft attend 는 hard 와 다르게, 하나만 고르지 않고, 모두 비율대로 고른다.
- Ex) hard 의 S_t = [0,0,1,0], soft 의 경우 [0.2, 0.1, 0.6, 0.1]
- $$\mathbb{E}_{p(s_t|a)}[\hat{\mathbf{z}}_t] = \sum_{i=1}^{L} \alpha_{t,i}\mathbf{a}_i$$
- $$\phi(\{\mathbf{a}_i\},\{\alpha_i\}) = \sum_i^L \alpha_i \mathbf{a}_i$$
- End to end 방법으로 그대로 계산해 주면 된다.

Doubly stochastic attention

- $\sum_i \alpha_{ti} = 1$ 이라는 조건은 모든 부분을 전체적으로 보는것을 방해
- $\sum_i \alpha_{ti} \approx 1$ 으로 약간 풀어주면 모든 부분을 전체적으로 보는것을 도와준다
- 그에 따라 더 focus 된 attention 이 가능해진다.

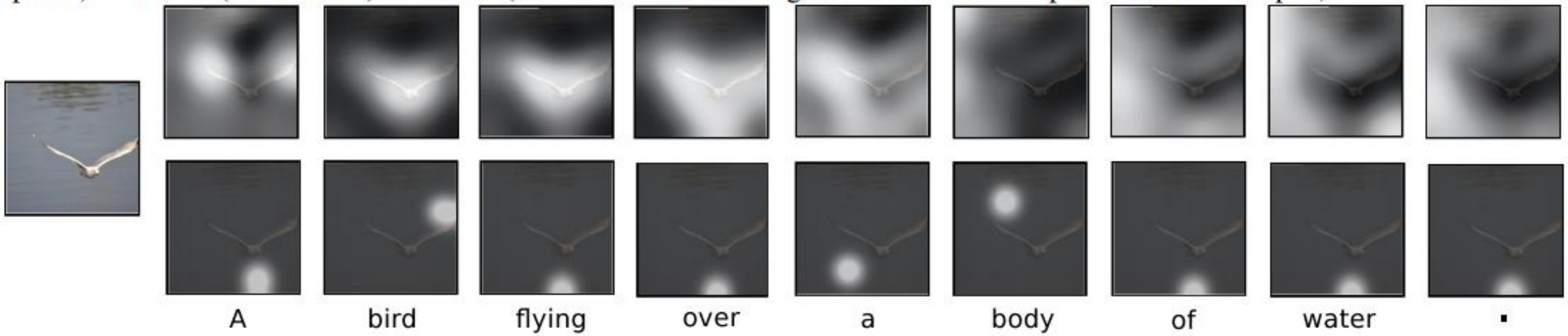
$$L_d = -\log(p(y|x)) + \lambda \sum_{i}^{L} \left(1 - \sum_{t}^{C} \alpha_{ti}\right)^2.$$

Pros and Cons

	Soft Attention	Hard Attention
Pros	• Interpretability • End to end learning	• Better performance than soft attention
Cons	• Worse performance than hard attention	• Hard to optimize - Monte Carlo based sampling - REINFORCE

Example

Figure 2. Attention over time. As the model generates each word, its attention changes to reflect the relevant parts of the image. "soft" (top row) vs "hard" (bottom row) attention. (Note that both models generated the same captions in this example.)



Example

Figure 3. Examples of attending to the correct object (*white* indicates the attended regions, *underlines* indicated the corresponding word)

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

A little girl sitting on a bed with a teddy bear.

A group of people sitting on a boat in the water.

A giraffe standing in a forest with trees in the background.

Encoder

Dataset	Model	BLEU				
		BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR
Flickr8k	Google NIC(Vinyals et al., 2014)$^{\dagger\Sigma}$	63	41	27	—	—
	Log Bilinear (Kiros et al., 2014a)$^{\circ}$	65.6	42.4	27.7	17.7	17.31
	Soft-Attention	**67**	44.8	29.9	19.5	18.93
	Hard-Attention	**67**	**45.7**	**31.4**	**21.3**	**20.30**
Flickr30k	Google NIC$^{\dagger\circ\Sigma}$	66.3	42.3	27.7	18.3	—
	Log Bilinear	60.0	38	25.4	17.1	16.88
	Soft-Attention	66.7	43.4	28.8	19.1	**18.49**
	Hard-Attention	**66.9**	**43.9**	**29.6**	**19.9**	18.46
COCO	CMU/MS Research (Chen & Zitnick, 2014)^{a}	—	—	—	—	20.41
	MS Research (Fang et al., 2014)$^{\dagger a}$	—	—	—	—	20.71
	BRNN (Karpathy & Li, 2014)$^{\circ}$	64.2	45.1	30.4	20.3	—
	Google NIC$^{\dagger\circ\Sigma}$	66.6	46.1	32.9	24.6	—
	Log Bilinear$^{\circ}$	70.8	48.9	34.4	24.3	20.03
	Soft-Attention	70.7	49.2	34.4	24.3	**23.90**
	Hard-Attention	**71.8**	**50.4**	**35.7**	**25.0**	23.04

Summary

- Encoder 은 CNN, Decoder 은 LSTM 을 사용한다.
- LSTM 은 전 caption 단어, 전 hidden state, attention model 이 생성하는 z 가 input 이 된다.
- Context vector z 는 Hard attention, soft attention 두가지 방법중 하나를 이용하게 된다.
- Hard attention 은 location variable s 를 정의하고, 이것을 이용해 likelihood 의 lower bound 를 계산하고 이를 maximize 하기위해 몬테카를로 샘플링을 이용한다.
- Soft attention 은 매 iter 마다 sampling 이 아닌, 직접 z 를 계산한다.
- Attention base model 은 기존 image caption model 보다 좋은 성능을 보였다.

Rerference

- http://dmqm.korea.ac.kr/activity/seminar/280
- https://ahjeong.tistory.com/8
- https://cool24151.tistory.com/71
- http://sanghyukchun.github.io/93/