CAP 6412 Advanced Computer Vision

http://www.cs.ucf.edu/~bgong/CAP6412.html

Boqing Gong

Feb 04, 2016

Today

- Administrivia
- Attention Modeling in Image Captioning, by Karan
- Neural networks & Backpropagation (Part VI)

Past due (02/04 Thursday, 12pm)

- Assignment 4: Review the following paper

{**Major**} Xu, Kelvin, Jimmy Ba, Ryan Kiros, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. “Show, attend and tell: Neural image caption generation with visual attention.” arXiv preprint arXiv:1502.03044 (2015).

Template for paper review:
http://www.cs.ucf.edu/~bgong/CAP6412/Review.docx

Next week:
CNN & Segmentation and super-resolution

Tuesday (02/09) Jose Sanchez	**[Super-resolution]** Dong, Chao, Chen Change Loy, Kaiming He, and Xiaoou Tang. "Learning a deep convolutional network for image super-resolution." In Computer Vision–ECCV 2014, pp. 184-199. Springer International Publishing, 2014. (Extended version on ArXiv) & Secondary papers
Thursday (02/11) Goran Igic	[**Edge detection**] Xie, Saining, and Zhuowen Tu. "Holistically-Nested Edge Detection." In Proceedings of the IEEE International Conference on Computer Vision, 2015. & Secondary papers

Project 1: Due in three weeks (02/28)

- If you choose option 2, your own project
  - Deadline for discussion & approval: 02/11/2016
  - See instructions on how to prepare the slides for discussion
  - http://www.cs.ucf.edu/~bgong/CAP6412/proj1.pdf

Today

- Administrivia
- Attention Modeling in Image Captioning, by Karan
- Neural networks & Backpropagation (Part VI)

Upload slides after class

- See “Paper Presentation” on UCF webcourse

- Sharing your slides
  - **Refer to the originals sources of images, figures, etc. in your slides**
  - Convert them to a PDF file
  - Upload the PDF file to “Paper Presentation” after your presentation

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

Kelvin Xu Jimmy Lei Ba Ryan Kiros
Kyunghyun Cho Aaron Courville Ruslan Salakhutdinov Richard S. Zemel
Yoshua Bengio

University of Montreal

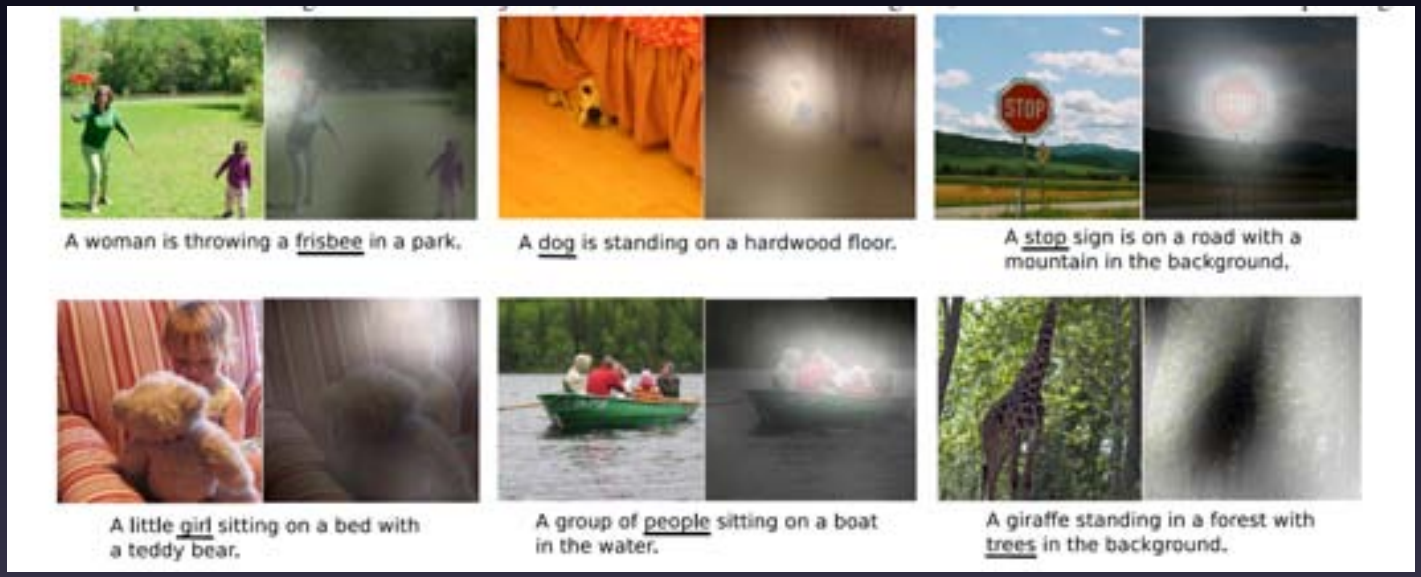
presented by Karan Daei-Mojdehi
February 04, 2016

Presentation Outline

- Motivation
- Problem Statement
- Approach
  - Big Picture of the Model
  - Encoder, Decoder, and Attention Model in the Big Picture
- Details: CNN as the Encoder
- Background on RNNs and LSTMS
- Getting into Details
  - LSTM as the Decoder
  - Soft Attention Model (Deterministic)
  - Hard Attention Model (Stochastic)
- Training Procedure
- Experiments & Results
- Future Directions

Motivation

- Scene Understanding as one of the primary goals of Computer Vision
- Mimicking human ability to compress huge visual information into descriptive language
- Automatic Image Captioning by exploiting attention model



$Source : KelvinXu$

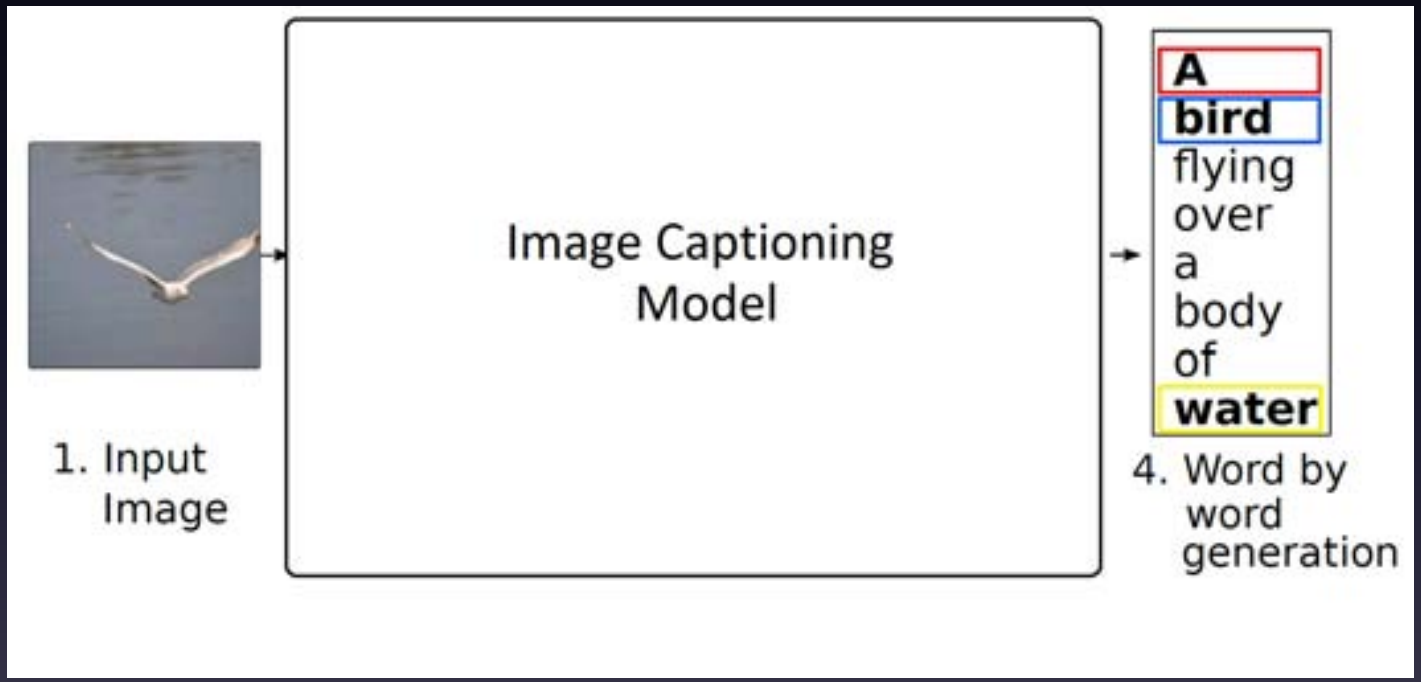
Problem Statement

- Given an input image, generate a set of words (caption) that best describe the image
- Caption is encoded as a sequence of 1-of-K encoded words($\mathbf{y_i}$) as output:

$$\text{caption} = \{\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_C\}$$
$$\mathbf{y}_i \in \mathbb{R}^{\mathbb{K}}$$

Approach: Big Picture

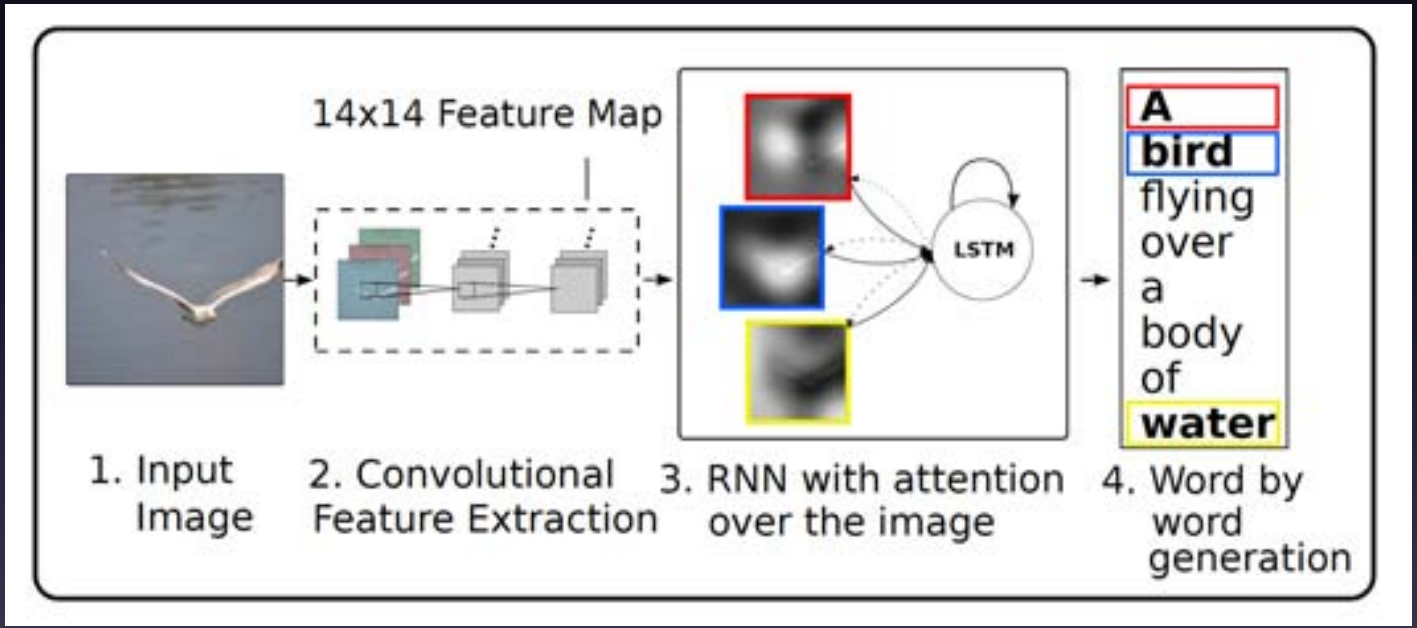
- Model should be able to generate a caption given a raw image:



Source : Kelvin Xu

Approach: Big Picture

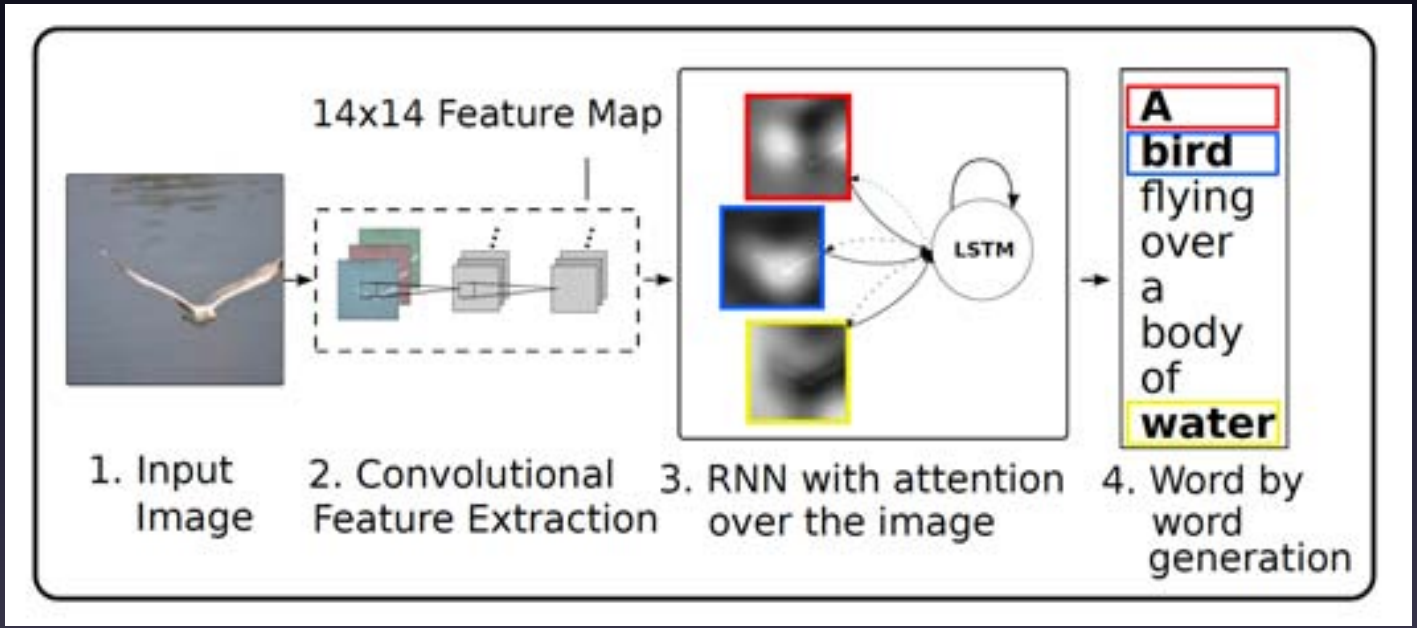
- a combination of CNN and RNN is used in an encoder/decoder manner
- how CNN and RNN are pipelined together is where attention model comes into play



Source : Kelvin Xu

Approach: Big Picture

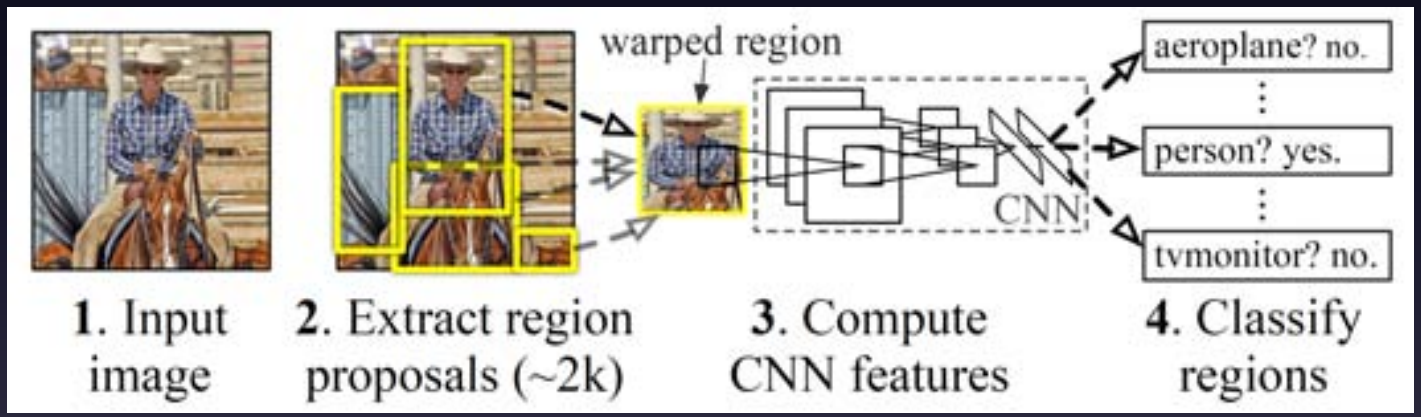
- a combination of CNN and RNN is used in an encoder/decoder manner
- how CNN and RNN are pipelined together is where attention model comes into play



Source : Kelvin Xu

Approach: Big Picture

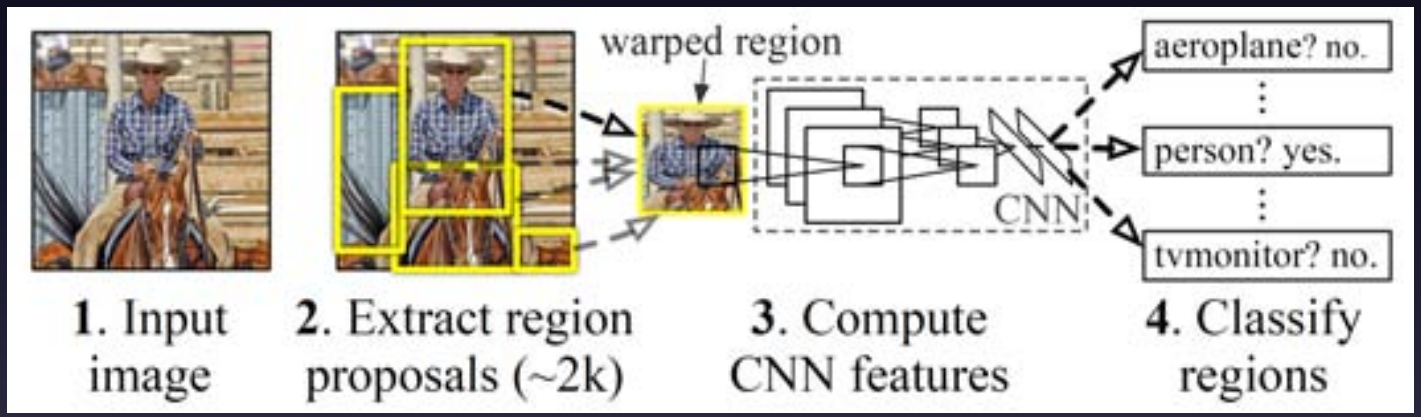
-
- compare to previous paper where object detection algorithms were used



Source : Kelvin Xu

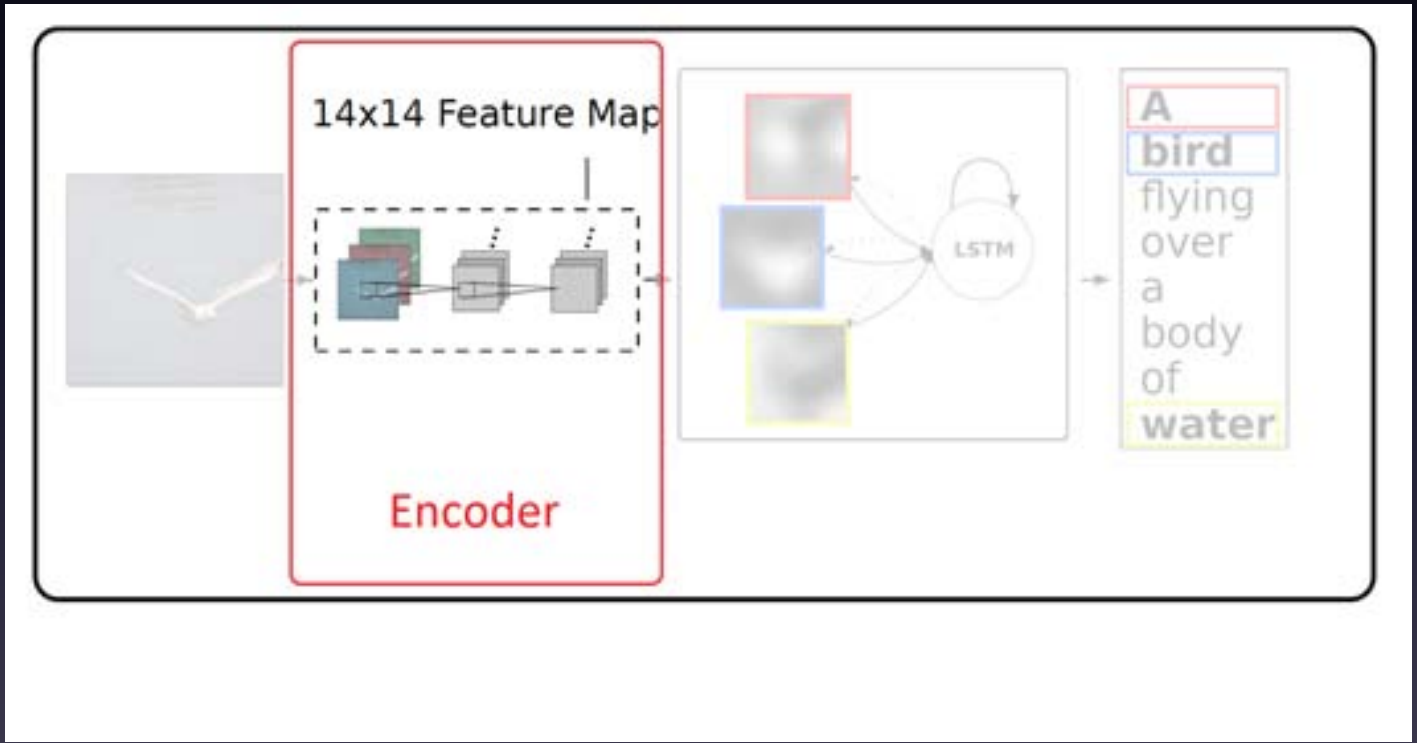
Approach: Big Picture

-
- compare to previous paper where object detection algorithms were used



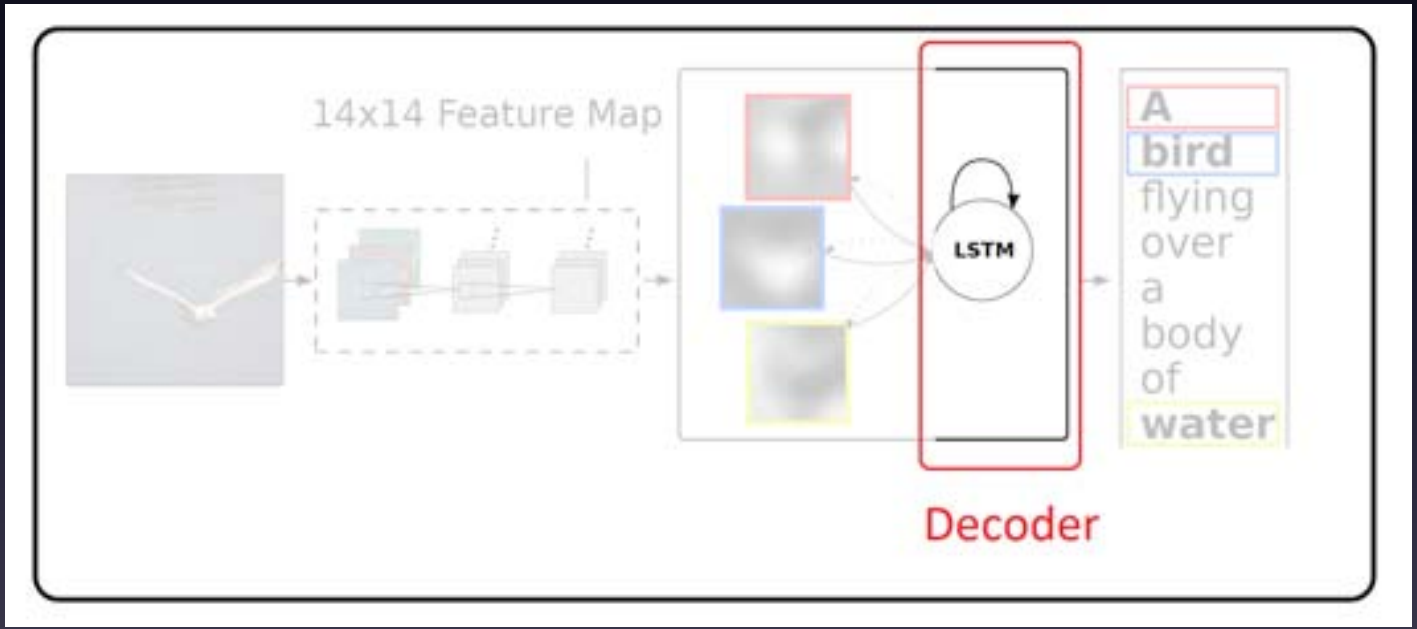
Source : Kelvin Xu

- Encoder is responsible for converting the image into feature vectors



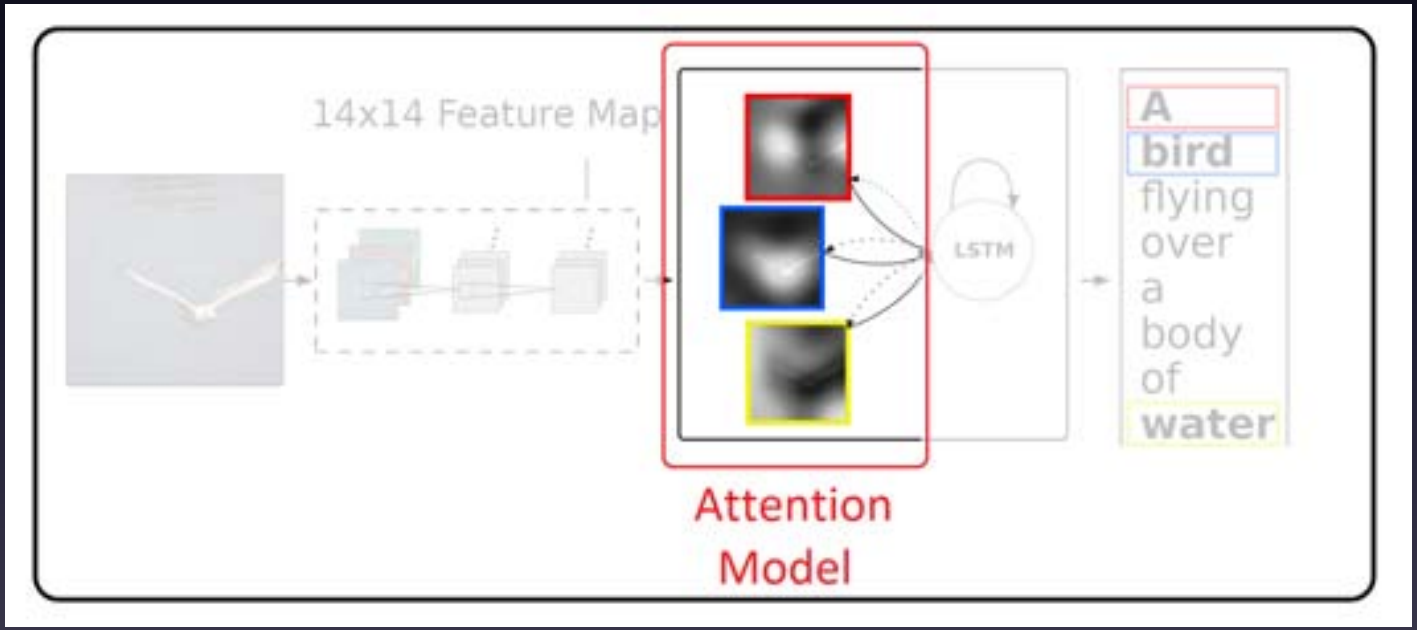
Source : Kelvin Xu

- Decoder generates captions
- Input to the decoder is a "selection" of set of feature vectors from the ecnoder



Source : Kelvin Xu

- decides "where" to look at each time step
- Attention Model determines which feature vector is fed into the decoder



Source : Kelvin Xu

Where We Are

- Motivation
- Problem Statement
- Approach
  - Big Picture of the Model
  - Encoder, Decoder, and Attention Model in the Big Picture
- Details: CNN as the Encoder
- Background on RNNs and LSTMS
- Getting into Details
  - LSTM as the Decoder
  - Soft Attention Model (Deterministic)
  - Hard Attention Model (Stochastic)
- Training Procedure
- Experiments & Results
- Future Directions

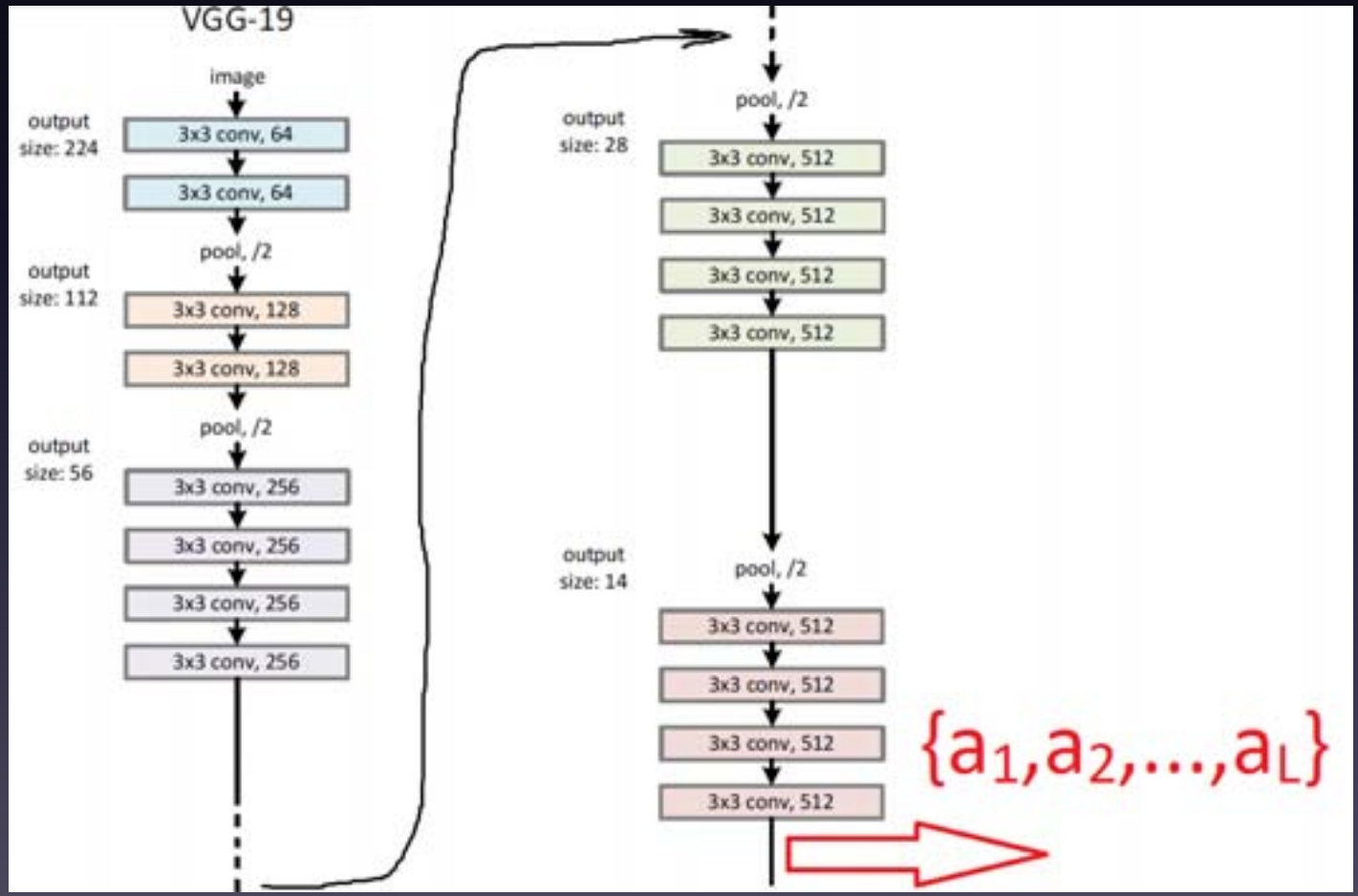
Details - CNN as the Encoder

- A CNN model is used to extract feature vectors from the image
- feature vectors are outputs from middle layers of a deep CNN
- we refer to set of these feature vectors as **annotation** vectors
- each annotation vector corresponds to a region of image

$$\textbf{annotation} = \{\mathbf{a}_1, \mathbf{a}_2, ..., \mathbf{a}_L\}$$
$$\mathbf{a}_i \in \mathbb{R}^{\mathbb{D}}$$

Details - CNN as the Encoder

- For experiments, the output after fourth maxpooling layer of Oxford VGGnet is selected as output of encoder:

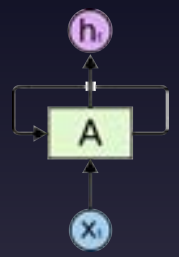


Where We Are

- Motivation
- Problem Statement
- Approach
  - Big Picture of the Model
  - Encoder, Decoder, and Attention Model in the Big Picture
- Details: CNN as the Encoder
- Background on RNNs and LSTMS
- Getting into Details
  - LSTM as the Decoder
  - Soft Attention Model (Deterministic)
  - Hard Attention Model (Stochastic)
- Training Procedure
- Experiments & Results
- Future Directions

Background on RNNs and LSTMs

- Recurrent Neural Networks are capable of handling variable length inputs and outputs
- => widely used in machine translation, image captioning, video captioning, etc
- this is gained by their capability of passing a vector (hiden state) in time



Imge Credits : Chris Olah

Background on RNNs and LSTMs

- A Recurrent Neural Network unrolled in time:

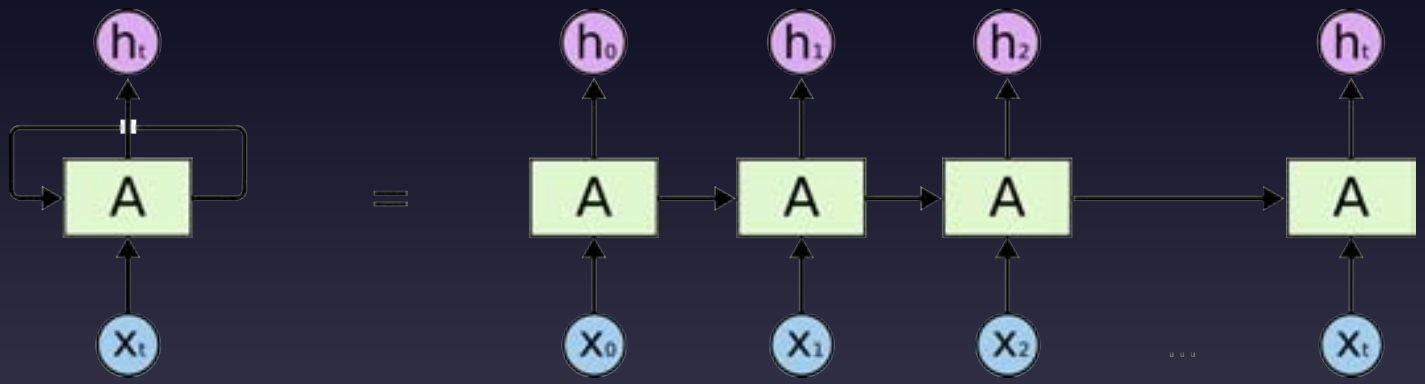


Image Credits : Chris Olah

Background on RNNs and LSTMs

A Toy example of Recurrent Neural Networks:

- Assume we want to generate the word: hello
- At training time, target is to predict next letter
- At execution time, given an init input, most probable next letter is sent as output and fed back to network as well

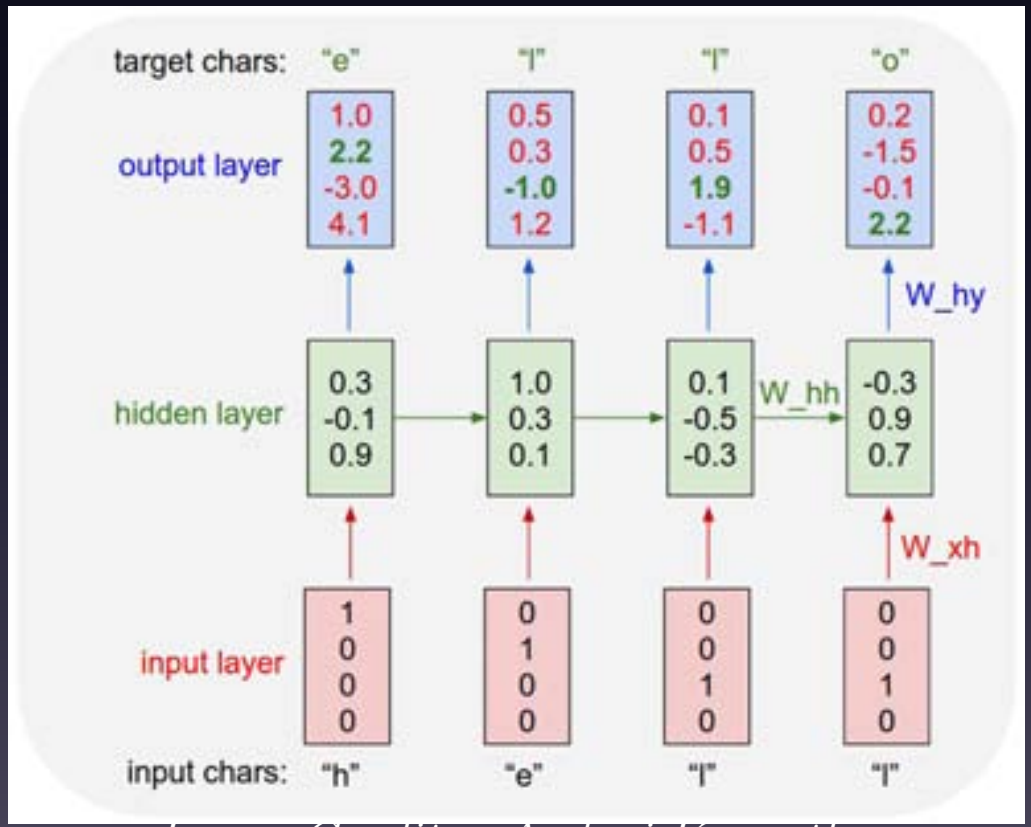
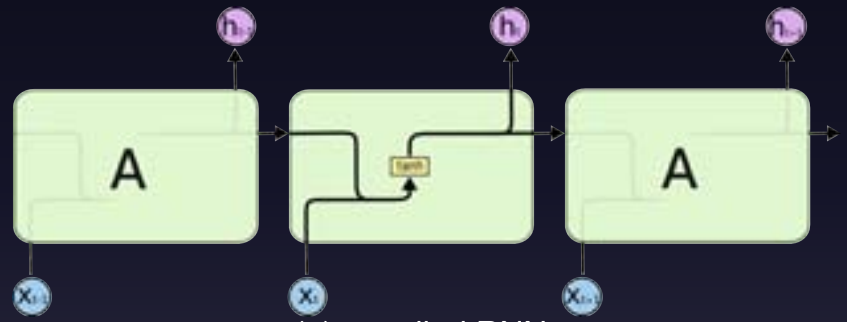


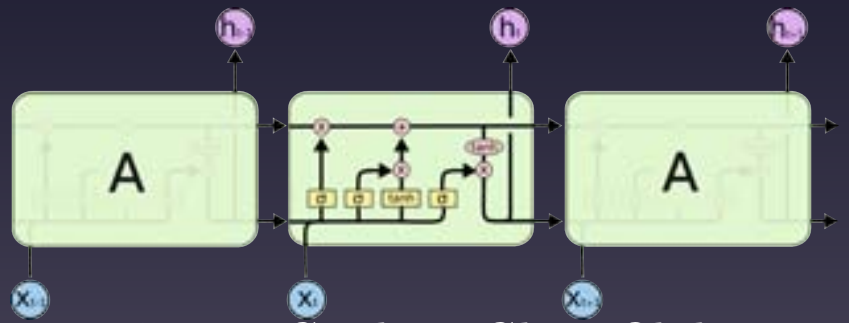
Image Credits : Andrej Karpathy

Background on RNNs and LSTMs

- Problem with RNNs: they become unable to learn to connect long term dependencies
- Solution: Long Short Term Memory recurrent networks (LSTMs)
- LSTMs have dynamics that allow them to hold information for long periods



(a) unrolled RNN



(b) unrolled LSTM

Image Credits: Chris Olah

Background on RNNs and LSTMs

Core idea behind LSTM dynamics...

- The key to LSTM is cell state
- cell state is capable of holding information long in time

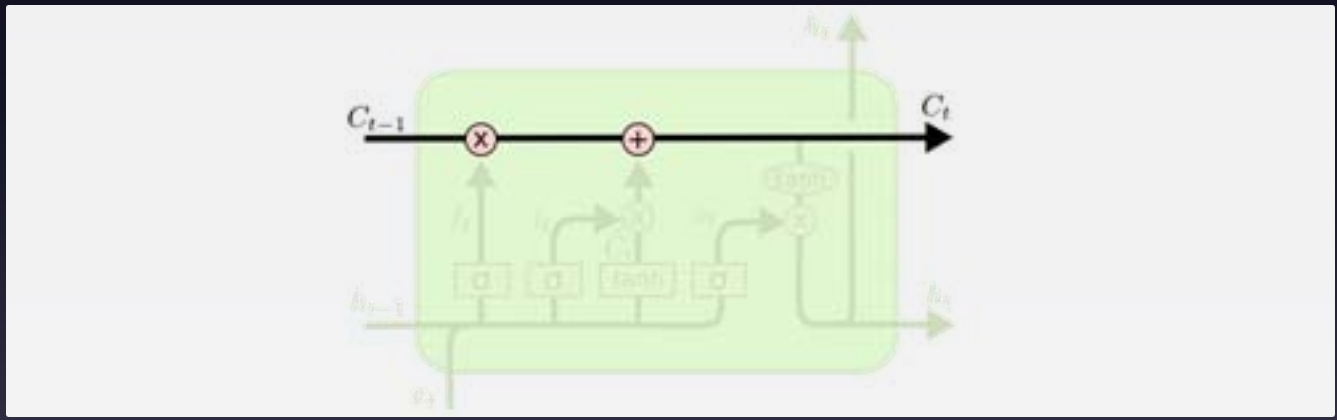


Image Credits : Chris Olah

Background on RNNs and LSTMs

Forget gate of LSTM:

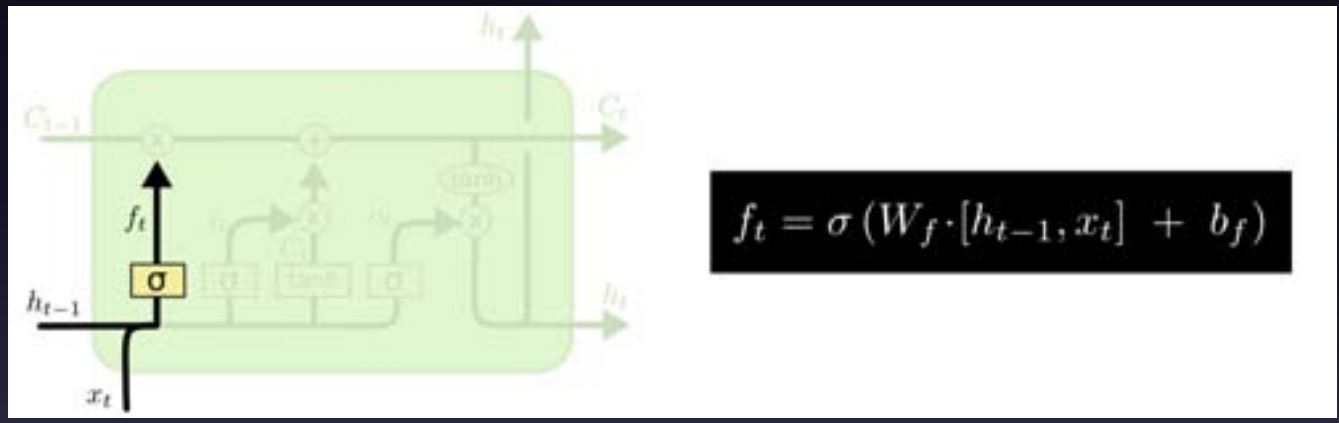


Image Credits : Chris Olah

Background on RNNs and LSTMs

Input gate of LSTM:

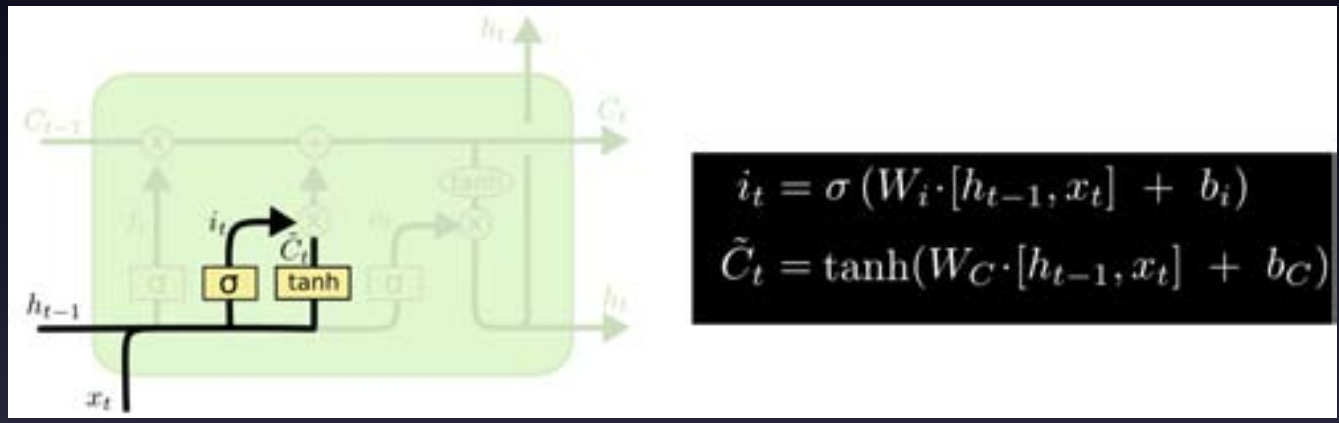


Image Credits : Chris Olah

Background on RNNs and LSTMs

Output (last) gate of LSTM:

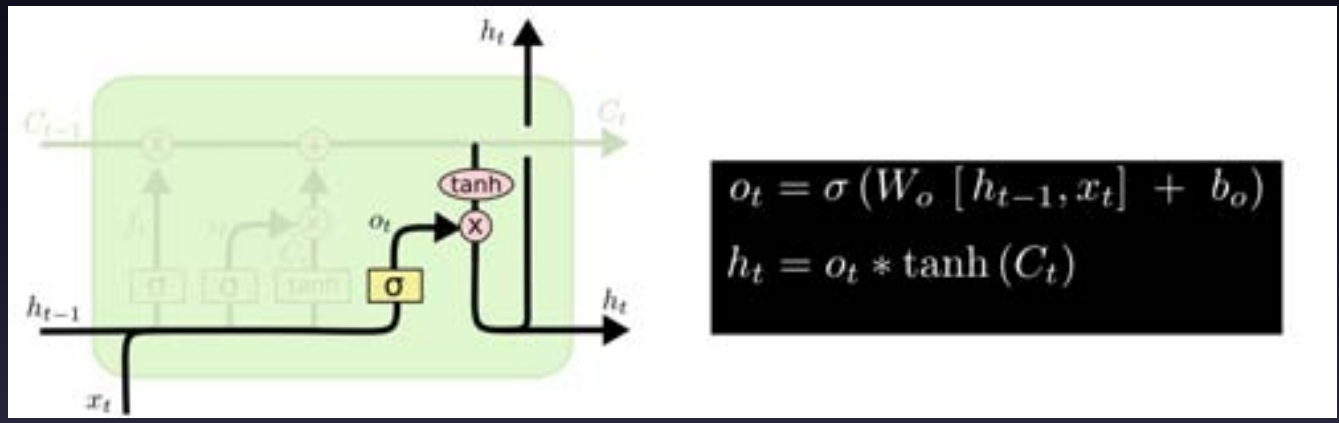


Image Credits : Chris Olah

Where We Are

- Motivation
- Problem Statement
- Approach
  - Big Picture of the Model
  - Encoder, Decoder, and Attention Model in the Big Picture
- Details: CNN as the Encoder
- Background on RNNs and LSTMS
- Getting into Details
  - LSTM as the Decoder
  - Soft Attention Model (Deterministic)
  - Hard Attention Model (Stochastic)
- Training Procedure
- Experiments & Results
- Future Directions

Details - LSTM as the Decoder

- Decoder in this paper is a model of LSTM
- inputs to this LSTM are:
  - vector z_t: corresponding to one of annotation vectors
  - vector h_{t-1} Previous state of LSTM
  - vector y_{t-1} Previous word generated by LSTM

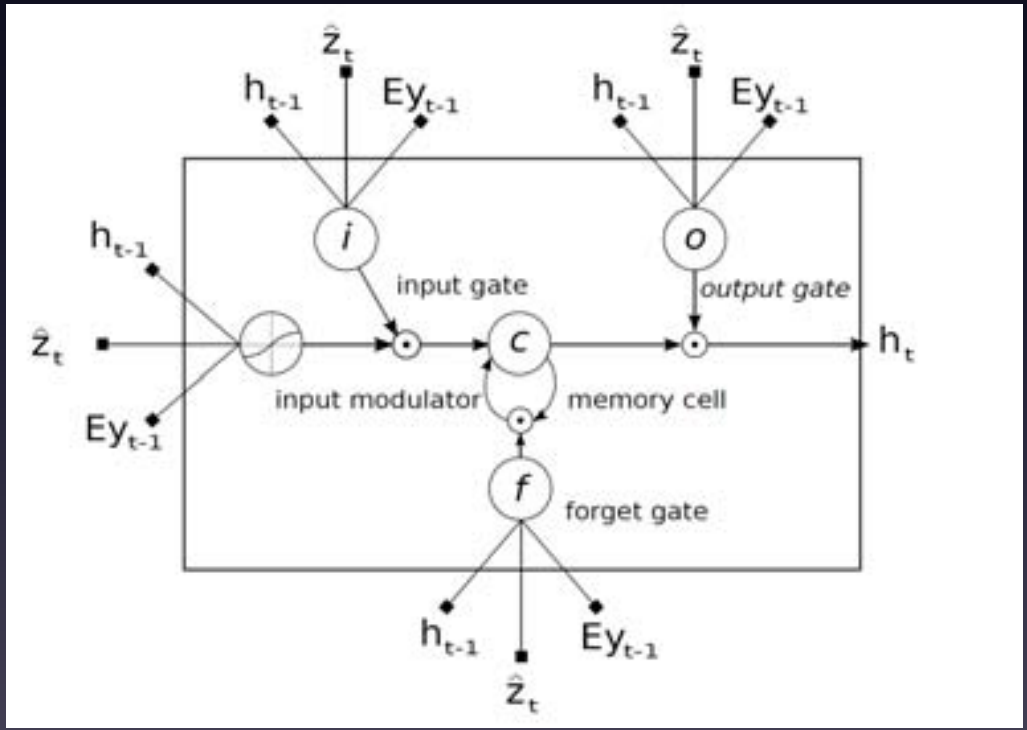


Image Credits : Kelvin Xu

Details - LSTM as the Decoder

- Output of this LSTM is a set of vectors corresponding to consequent words (caption)
- A deep ouptut layere for this LSTM is designed as follows:
- Shows the probability of next word, given current state, previous output, and region of image where attention is upon

$$p(\mathbf{y}_t|\mathbf{a}, \mathbf{y}_1^{t-1}) \propto \exp(\mathbf{L}_o(\mathbf{E}\mathbf{y}_{t-1} + \mathbf{L}_h\mathbf{h}_t + \mathbf{L}_z\hat{\mathbf{z}}_t))$$

Details - Attention Model

Recall:

- Annotation vectors as output of Encoder (CNN):

$$\textbf{annotation} = \{\mathbf{a}_1, \mathbf{a}_2, ..., \mathbf{a}_L\}$$
$$\mathbf{a}_i \in \mathbb{R}^{\mathbb{D}}$$

Details - Attention Model

Recall:

- Annotation vectors as output of Encoder (CNN):

$$\textbf{annotation} = \{\mathbf{a}_1, \mathbf{a}_2, ..., \mathbf{a}_L\}$$
$$\mathbf{a}_i \in \mathbb{R}^{\mathbb{D}}$$

- And one of inputs to our Decoder (LSTM) was z_t

Details - Attention Model

Recall:

- Annotation vectors as output of Encoder (CNN):

$$\textbf{annotation} = \{\mathbf{a}_1, \mathbf{a}_2, ..., \mathbf{a}_L\}$$
$$\mathbf{a}_i \in \mathbb{R}^{\mathbb{D}}$$

- And one of inputs to our Decoder (LSTM) was z_t
- z_t was supposed to correspond to a region of image where "attention" was upon

Details - Attention Model

Recall:

- Annotation vectors as output of Encoder (CNN):

$$\textbf{annotation} = \{\mathbf{a}_1, \mathbf{a}_2, ..., \mathbf{a}_L\}$$
$$\mathbf{a}_i \in \mathbb{R}^{\mathbb{D}}$$

- And one of inputs to our Decoder (LSTM) was z_t
- z_t was supposed to correspond to a region of image where "attention" was upon
- Attention Model is responsible to make the connection between these two

Details - Attention Model

- This mechanism is done by first assigning a factor $\alpha_{ti} i = 1, 2, .., L$ to each annotation vector

Details - Attention Model

- This mechanism is done by first assigning a factor $\alpha_{ti} i = 1, 2, .., L$ to each annotation vector
- We can think of α_{ti} as:
  - either relative importance to give to annotation vector i at current time t (soft attention)

Details - Attention Model

- This mechanism is done by first assigning a factor $\alpha_{ti} i = 1, 2, .., L$ to each annotation vector
- We can think of α_{ti} as:
  - either relative importance to give to annotation vector i at current time t (soft attention)
  - or probability of ith annotation vector being the correct region for current time step

Details - Attention Model

- This mechanism is done by first assigning a factor $\alpha_{ti} i = 1, 2, .., L$ to each annotation vector
- We can think of α_{ti} as:
  - either relative importance to give to annotation vector i at current time t (soft attention)
  - or probability of ith annotation vector being the correct region for current time step
- in either model, each α_{ti} is learnt by applying a softmax to an MLP:

$$\alpha_{ti} = \frac{exp(e_{ti})}{\sum_{k=1}^{L} exp(e_{tk})}$$

Details - Attention Model

- This mechanism is done by first assigning a factor $\alpha_{ti} i = 1, 2, .., L$ to each annotation vector
- We can think of α_{ti} as:
  - either relative importance to give to annotation vector i at current time t (soft attention)
  - or probability of ith annotation vector being the correct region for current time step
- in either model, each α_{ti} is learnt by applying a softmax to an MLP:

$$\alpha_{ti} = \frac{exp(e_{ti})}{\sum_{k=1}^{L} exp(e_{tk})}$$

- where e_{ti} is learnt by an MLP conditioned on previous hidden states (f_{att})

$$e_{ti} = f_{att}(a_i, h_{t-1})$$

Details - Attention Model

- This mechanism is done by first assigning a factor $\alpha_{ti} i = 1, 2, .., L$ to each annotation vector
- We can think of α_{ti} as:
  - either relative importance to give to annotation vector i at current time t (soft attention)
  - or probability of ith annotation vector being the correct region for current time step
- in either model, each α_{ti} is learnt by applying a softmax to an MLP:

$$\alpha_{ti} = \frac{exp(e_{ti})}{\sum_{k=1}^{L} exp(e_{tk})}$$

- where e_{ti} is learnt by an MLP conditioned on previous hidden states (f_{att})

$$e_{ti} = f_{att}(a_i, h_{t-1})$$

- i.e. where we look depends on current state of LSTM

Details - Attention Model

- having α_{ti}s all we need is a mechanism Φ that selects appropriate image region(s):

$$z_t = \Phi(\{a_i\}, \{\alpha_i\})$$

Details - Soft Attention Model

In soft attention model, Φ simply outputs expected value of annotation vectors:

$$z_t = \sum_{i=1}^{L} \alpha_{t,i}.a_i$$

Details - Hard Attention Model

- in hard attention model, sample are stochastic ally drawn from a multinouli distribution

Details - Hard Attention Model

- in hard attention model, sample are stochastic ally drawn from a multinouli distribution
- WHAT DOES THAT MEAN!?

Details - Hard Attention Model

- in hard attention model, sample are stochastic ally drawn from a multinouli distribution
- WHAT DOES THAT MEAN!?
- we simulate different scenarios in each, a specific image region is fed to our LSTM

Details - Hard Attention Model

- in hard attention model, sample are stochastic ally drawn from a multinouli distribution
- WHAT DOES THAT MEAN!?
- we simulate different scenarios in each, a specific image region is fed to our LSTM
- the probability of these scenarios are weighted by our α_{ti}s at each iteration

Where We Are

- Motivation
- Problem Statement
- Approach
  - Big Picture of the Model
  - Encoder, Decoder, and Attention Model in the Big Picture
- Details: CNN as the Encoder
- Background on RNNs and LSTMS
- Getting into Details
  - LSTM as the Decoder
  - Soft Attention Model (Deterministic)
  - Hard Attention Model (Stochastic)
- Training Procedure
- Experiments & Results
- Future Directions

Training Procedure

after creating the pipeline encoder, decoder through the attention model:

- a pre-trained CNN (VGGnet) on ImageNet is used and fixated

Training Procedure

after creating the pipeline encoder, decoder through the attention model:

- a pre-trained CNN (VGGnet) on ImageNet is used and fixated
- in the training phase, one word of each image caption is fed to the pipeline at each time step

Training Procedure

after creating the pipeline encoder, decoder through the attention model:

- a pre-trained CNN (VGGnet) on ImageNet is used and fixated
- in the training phase, one word of each image caption is fed to the pipeline at each time step
- target is maximizing the probability of pipeline output to be next word in the caption

Training Procedure

after creating the pipeline encoder, decoder through the attention model:

- a pre-trained CNN (VGGnet) on ImageNet is used and fixated
- in the training phase, one word of each image caption is fed to the pipeline at each time step
- target is maximizing the probability of pipeline output to be next word in the caption
- at each iteration, the error is backpropagated in all the modules except the CNN

Experiments & Results

Results from running the model on Flickr8K, Flickr30k, and MSCOCO using BLEU and METEOR metrics

- Flickr8k and Flickr30k both come with 5 reference captions per image
- MSCOCO exceeds this number for some images, (discarded for consistency)

		BLEU				
Dataset	Model	B-1	B-2	B-3	B-4	METEOR
Flickr8k	Google NIC(Vinyals et al., 2014)†Σ	63	41	27	—	—
	Log Bilinear (Kiros et al., 2014a)°	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†°Σ	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)†ª	—	—	—	—	20.71
	BRNN (Karpathy & Li, 2014)°	64.2	45.1	30.4	20.3	—
	Google NIC†°Σ	66.6	46.1	32.9	24.6	—
	Log Bilinear°	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

Source : Chris Olah

Experiments & Results

Visualization Results from the hard attention model

Source : Chris Olah

"A man and a woman playing frisbee in a field"

Experiments & Results

Visualization Results from the soft attention model (compare bright regions with hard model)

Source : Chris Olah

"A woman is throwing a frisbee in a park"

Future Directions

The framework of this paper can be used in other areas for combining different machine learning tools such as:

- in previous works where recurrent neural networks were used to detect multiple objects in a single image by looking at different regions of image
- Object tracking is another application that by intuition we can tell will benefit from this technique.
- Another area that this might perform well is action recognition in videos
- I might also investigate the effect of using attention for estimating relative depth in single images

Done!

Thank you for your attention

Questions

Just in case I make it this far...

- Any Questions?