

## Lab2 : Caption generation with visual attention

### Lab Objective:

In this lab, you are going to run a Caption generator by using CNN and RNN language generator trained on the COCO dataset to generate a sentence that describes the image.

### Important Date:

1. Submit Experiment Report Deadline: 11/21 (Tue) 12:00
2. Demo date: 11/22 (Wed) after Lab

### Requirements:

- Use ResNet101 as a pre-trained CNN model.
- Implement attention model in Show, attend and tell [1]
- Implement attention model in Bottom-Up and Top-Down Attention [3]
- Compare the result of attention mechanisms. (Loss curves, output captions, attention maps)

### Environment:

- COCO dataset

The COCO dataset is a large-scale object detection, segmentation, and captioning dataset. It consists of 330K images with 5 captions per image.

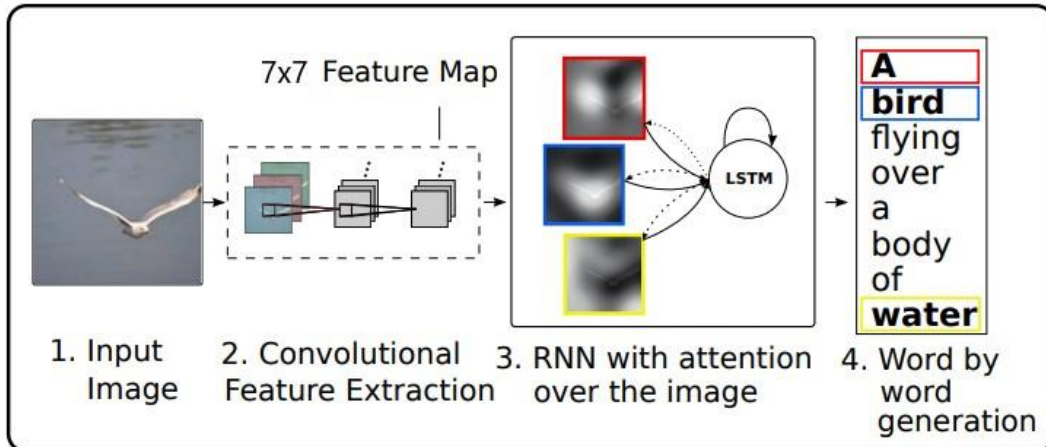
a man looking at a motorcycle with other motorcycles behind him.  
men looking at custom motorcycles in a motorcycle show  
a motorcycle sitting on a lush green lawn.  
a very nice motorcycle among others and a crowd of people.  
an elegantly painted custom motorcycle parked on the grass.



- Project package location: /homework/AI\_Training\_1108

### Lab Description:

- Sample code: <https://github.com/ruotianluo/neuraltalk2.pytorch>
- Learn how to combine CNN features and RNN language generator with attention model.
- The main structure of caption generator with attention is shown below:



- Attention model
  - Give the feature map weights to increase the importance of features.



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



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

- Context vector (output of attention model)

$$C_t = \sum_{i=1}^L \alpha_{ti} V_i$$

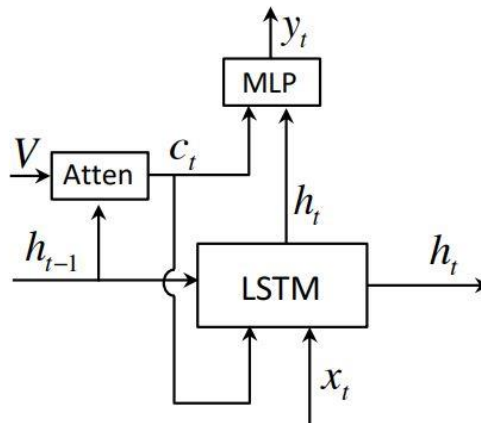
- Positive weight

$$\alpha_{ti} = \text{softmax}(e_{ti}) = \exp(e_{ti}) / \sum_k \exp(e_{tk})$$

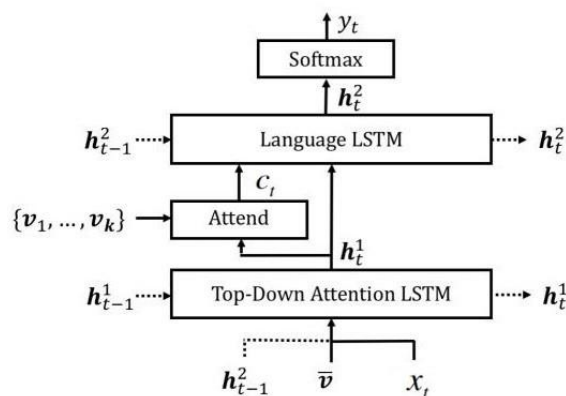
- Attention

$$e_t = \mathbf{W} \tanh(\mathbf{W}_v \mathbf{V} + \mathbf{W}_h \mathbf{h}_{t-1})$$

- Network architecture
  - Show, attend and tell



- Bottom-Up and Top-Down Attention
  - ◆ Separate the LSTM into Attention part and Language part

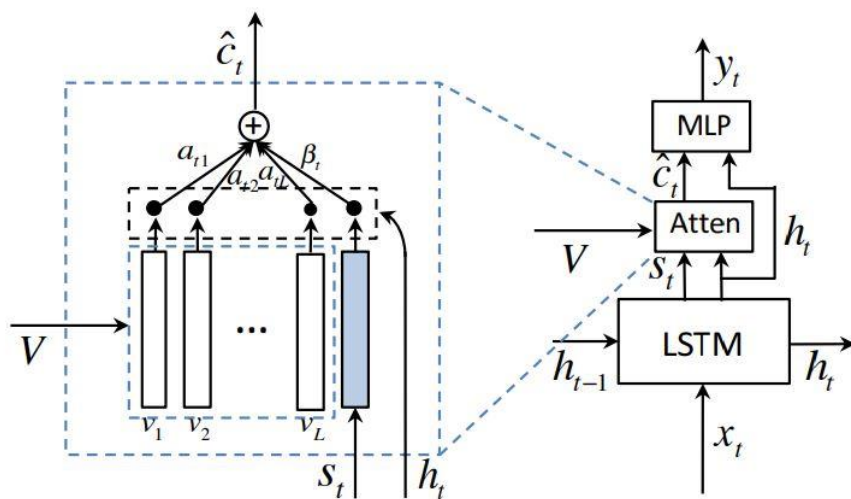


- Before training, you need to:
  - ~~Download coco images (2014 train images/ 2014 val images)~~
    - ◆ ~~<http://cocodataset.org/#download>~~
    - ◆ ~~Put them in the same directory~~
  - ~~Clone coco caption evaluation codes from GitHub~~
    - ◆ ~~<https://github.com/tylin/coco-caption>~~
  - Update the code from python 2.7 to 3 by \$ 2to3 -W -n ./< Folder>
  - Preprocess dates (process json dataset into hdf5/json files)
    - ◆ scripts/prepro\_images.py
    - ◆ scripts/prepro\_labels.py
  - Fill out the code in TODO field
    - ◆ misc/util.py (using pre-trained ResNet w/o avg. pooling and fc layer)
    - ◆ models/CaptionModel.py (for Show, attend and tell)
    - ◆ models/AttModel.py (for Bottom-Up and Top-Down Attention)

- Training
  - Run at least 5 epoch.
  - Training parameters
    - ◆ batch size : 10
    - ◆ input encoding size : 512
    - ◆ rnn size : 512
    - ◆ att hid size: 512
    - ◆ fc feat size: 2048
    - ◆ att feat size: 2048
    - ◆ rnn type: LSTM
  - ... see more setting of parameters in opt.py
  - Training Settings
    - ◆ --caption\_model <model name> : choose a caption model
    - ◆ --checkpoint\_path <folder> : the directory to save models
- Training Time: approx. 35mins per epoch on GTX1080 Ti

### Extra Bonus:

- Implement attention model in Knowing When to Look [4]
  - Adaptive attention model

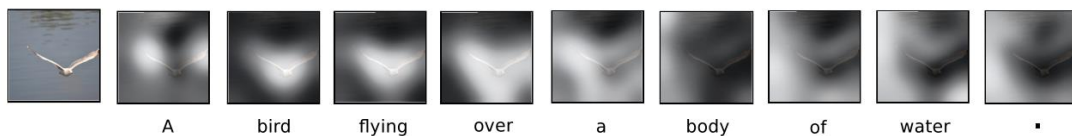


## References:

- [1] Xu, Kelvin, et al. "Show, attend and tell: Neural image caption generation with visual attention." *International Conference on Machine Learning*. 2015.
- [2] <http://people.ee.duke.edu/~lcarin/Yunchen9.25.2015.pdf>
- [3] Anderson, Peter, et al. "Bottom-up and top-down attention for image captioning and vqa." *arXiv preprint arXiv:1707.07998* (2017).
- [4] Lu, Jiasen, et al. "Knowing when to look: Adaptive attention via A visual sentinel for image captioning." *arXiv preprint arXiv:1612.01887* (2016).
- [5] <https://github.com/ruotianluo/neuraltalk2.pytorch>

## Report Spec: [black: Demo, Gray: No Demo]

1. Introduction (15%, 15%)
2. Experiment setup (15%, 15%)
  - The detail of your model
  - Report all your training hyper-parameters
3. Result (30%, 40%)
  - Training loss of attention models.
  - Caption of models.
  - Attention over time.



4. Discussion (20%, 30%)

Demo (20%) [抽人]

----- Criterion of result (Show\_attend\_tell)-----

Minimum loss < 1.8 = 100%

Minimum loss < 1.9 = 90%

Minimum loss < 2.0 = 80%

Minimum loss < 2.2 = 70%

Minimum loss > 2.2 = 0%

評分標準: 40%\*實驗結果 + 60%\*(報告+DEMO)