CNN Applications

CS5242

Lee Hwee Kuan & **Wang Wei**

Teaching assistant:

Connie Kou Khor Li, Ji Xin, Ouyang Kun

cs5242@comp.nus.edu.sg

Recap

- Operations
  - Convolution
  - Pooling
  - Dropout
  - ReLU
  - Batch-Normalization
  - Fully Connected
  - Softmax
  - Cross-entropy

Recap

- Architectures for ImageNet classification
  - AlexNet
  - VGG
  - InceptionNet
  - ResNet

Roadmap

CNN applications

- Image classification
- Object detection
- Object segmentation

Attention modelling

Image classification

- Predict the class/label of the image
- Training label
  - ground truth label (index)
- Test output
  - A probability distribution vector, one probability per label

Source from [13]

Training label: bicycle
Prediction output:
bicycle 0.6; people 0.3; mountain 0.05;

Image classification

- Approaches
  - AlexNet, VGG, InceptionNet, ResNet, DenseNet, etc
  - With a Softmax layer as the final output layer
  - With cross-entropy as the loss function
- Dataset
  - ImageNet
- Evaluation
  - Top-1: accuracy = #(top1 prediction is truth label) / # test samples
  - Top-5: accuracy = #(one of top5 prediction is truth label) / # test samples

Applications

- Logo classification
- Traffic sign classification
  - notebook
- Ecommerce product classification
- Medical image classification
- Food image classification
- ImageNet classification
- Dogs vs Cats
  - notebook

Image annotation

- Approaches
  - Same architecture as image classification
  - Binary classification for each label (0 vs 1)
    - Logistic function as the output layer
    - Cross-entropy for each label
  - Other output and loss layers [1]
- Evaluation [1]
  - Precision = average over all test samples {|Prediction ∩ Truth| / #Prediction}
  - Recall = average over all test samples {|Prediction ∩ Truth| / #Truth}

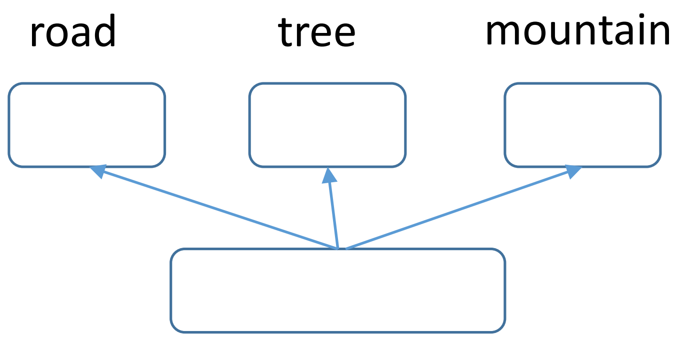


Image annotation

- Application
  - Example: [satellite image annotation]
  - [Notebook]

Source from: https://www.kaggle.com/c/planet-understanding-the-amazon-from-space

Object detection

- Detect the location of all object instances of all classes
- Training label
  - a list of <class, bounding box of each object instance>
- Prediction output
  - a list of <class, probability, bounding box of each object instance>

Training label:
bicycle (10, 100, 110, 110) (200, 200, 180, 80) …
People (200, 80, 71, 71) (300, 50, 20, 80) …

Prediction output:
bicycle 0.9 (9, 93, 100, 111), 0.8 (200, 200, 180, 80), …
people 0.8 (200, 80, 71, 71), …
….

Object detection

- Applications
  - Face detection
    - Point-and-shoot camera
  - Surveillance
    - Count cars, peoples, animals
  - Indexing
    - Get objects from images for search

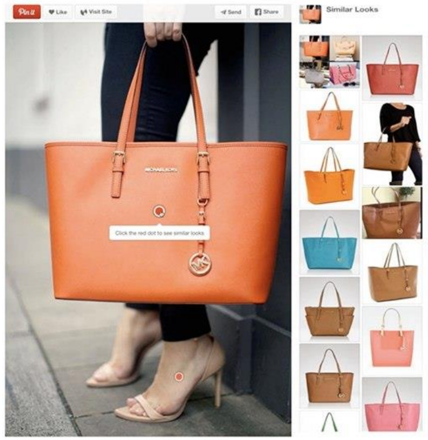


Figure 1: Similar Looks: We apply object detection to localize products such as bags and shoes. In this prototype, users click on automatically tagged objects to view similar-looking products.

Source from [10]

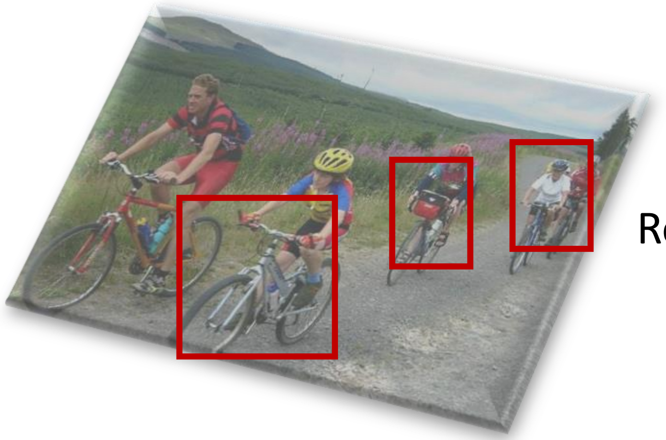
- Evaluation
  - Matched prediction = detected bounding box has enough overlap with truth and its label is correct
  - Precision = #matched prediction / #total predictions
  - Recall = #matched prediction / #truth instance (bounding box)

Object detection

- Solution
  - Find some candidate regions with objects
  - Extract CNN feature from this region
  - Refine the region boundary (bounding box) using a regressor
    - Generate 4 values (x, y, h, w)
  - Predict the class label using Softmax

R-CNN [7]

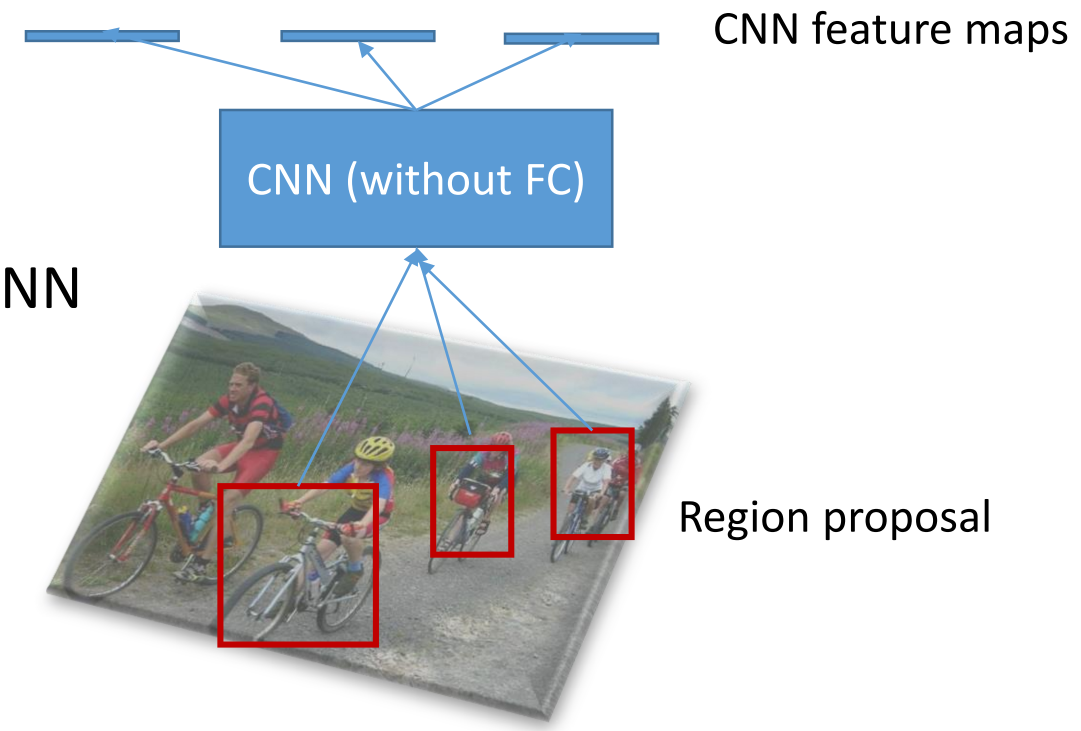
- Generate region proposal
  - Candidate object regions
  - Using existing methods



Region proposal

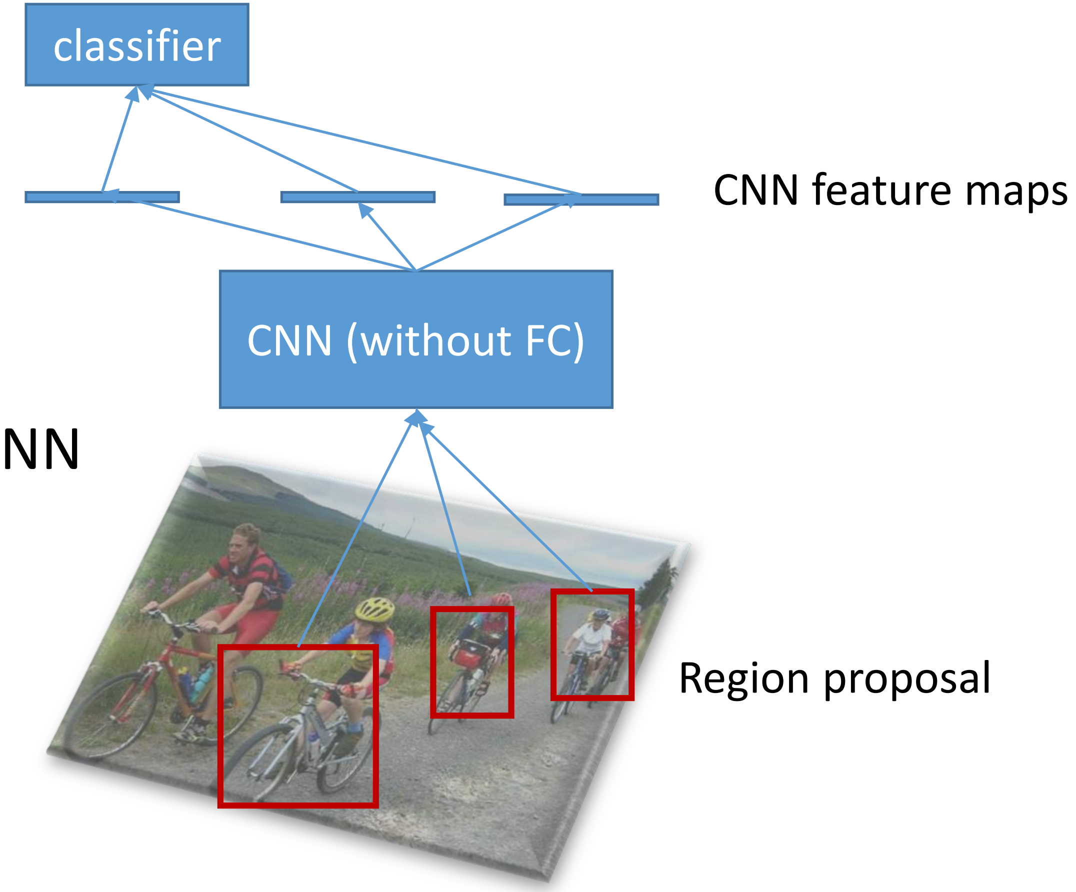
R-CNN

- Generate region proposal
  - Candidate object regions
  - Using existing methods
- Extract CNN feature
  - For each region
  - Forward-propagate each region via CNN
  - Using popular CNN architecture
    - VGG/ResNet/etc



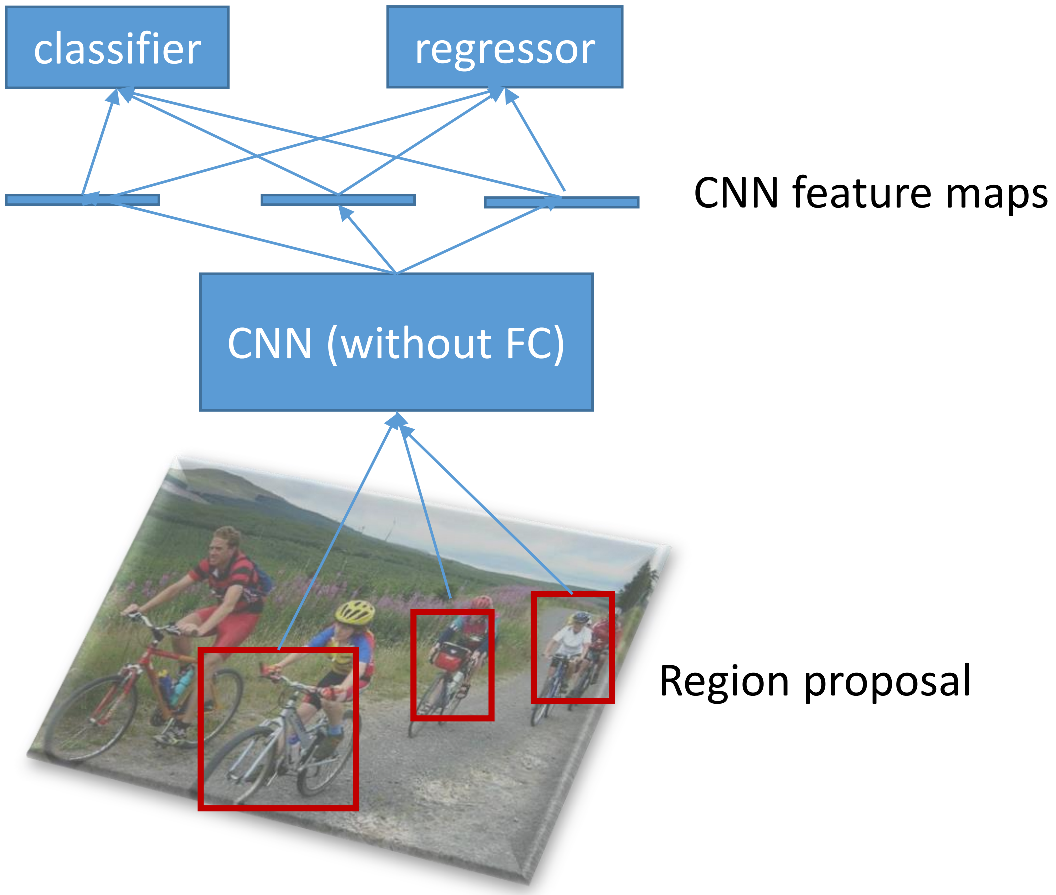
R-CNN

- Generate region proposal
  - Candidate object regions
  - Using existing methods
- Extract CNN feature
  - For each region
  - Forward-propagate each region via CNN
- Predict label for each region
  - Like image classification
  - Linear layer + softmax



R-CNN

- Generate region proposal
  - Candidate object regions
  - Using existing methods
- Extract CNN feature
  - For each region
  - Forward-propagate each region via CNN
- Predict label for each region
  - Like image classification
  - Linear layer + softmax
- Regress bounding box for each region
  - Linear regression for each value
  - 4 values (coordinates, or x,y,h,w)

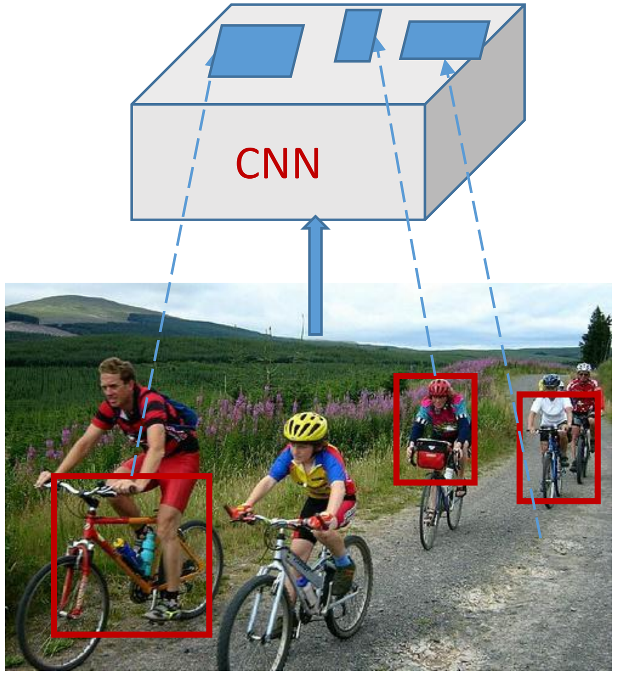


R-CNN

- Slow
  - Too many region proposals~2000
  - Each has to go through the CNN
- Training is ad-hoc
  - Fine tune the CNN for the target dataset for image classification
  - Train label classifier for regions
  - Train SVM regressor for bounding box

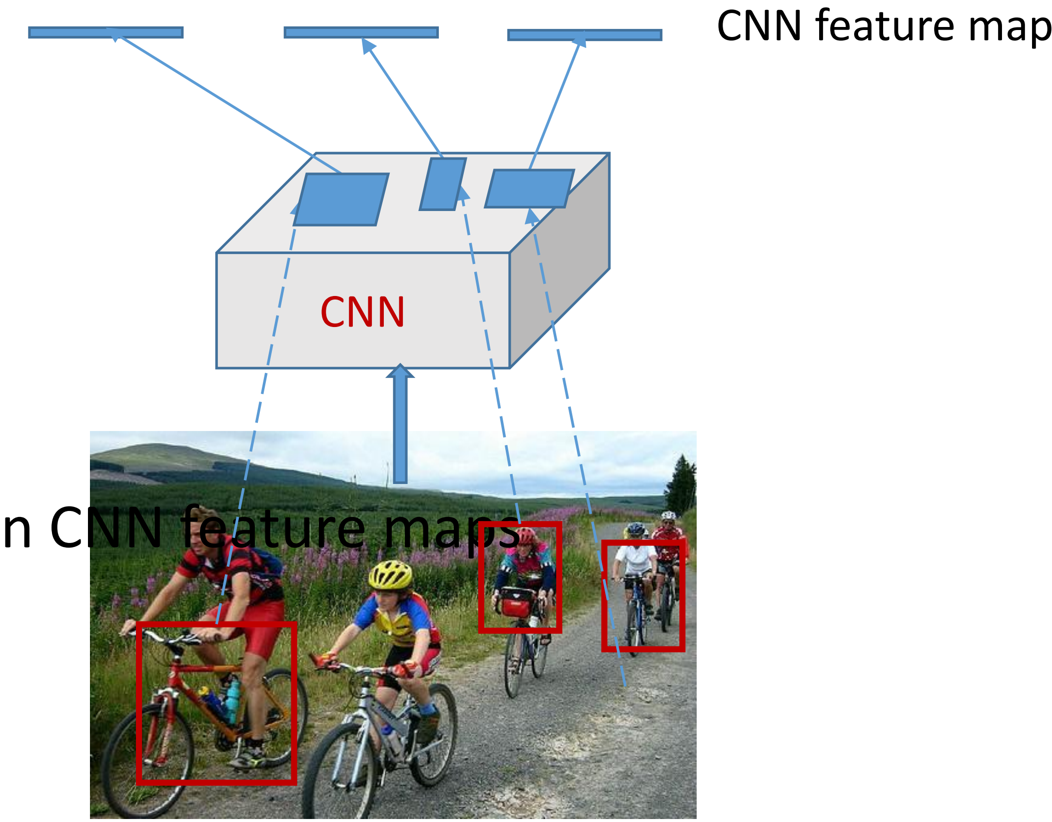
Fast R-CNN[2]

- Generate region proposal
  - Candidate object regions
  - Using existing methods
- Extract CNN feature
  - For the **whole input image**
  - Forward-propagate the whole image via CNN



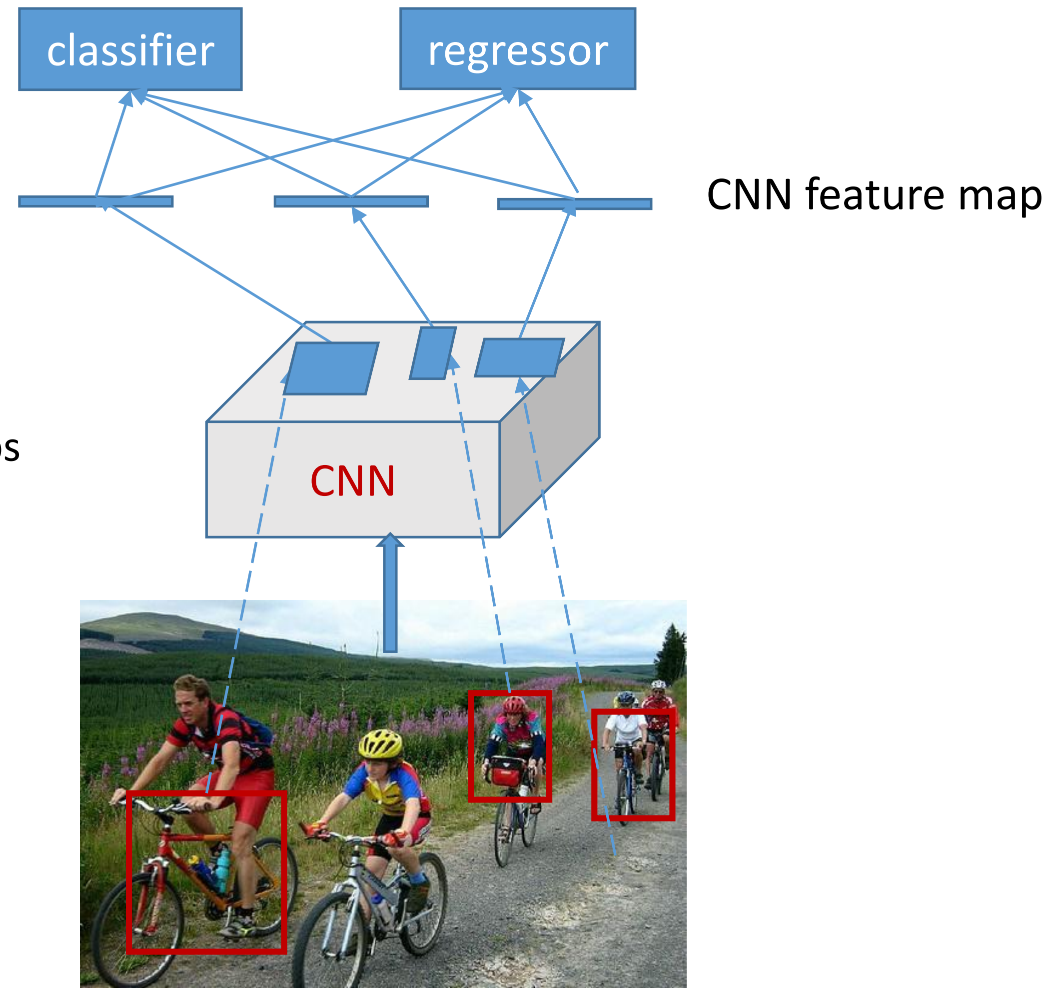
Fast R-CNN

- Generate region proposal
  - Candidate object regions
  - Using existing methods
- Extract CNN feature
  - For the **whole input image**
  - Forward-propagate the whole image via CNN
- Get CNN feature for each region
  - Using the region coordinates to locate areas in CNN feature maps
  - Extract the CNN feature from the areas



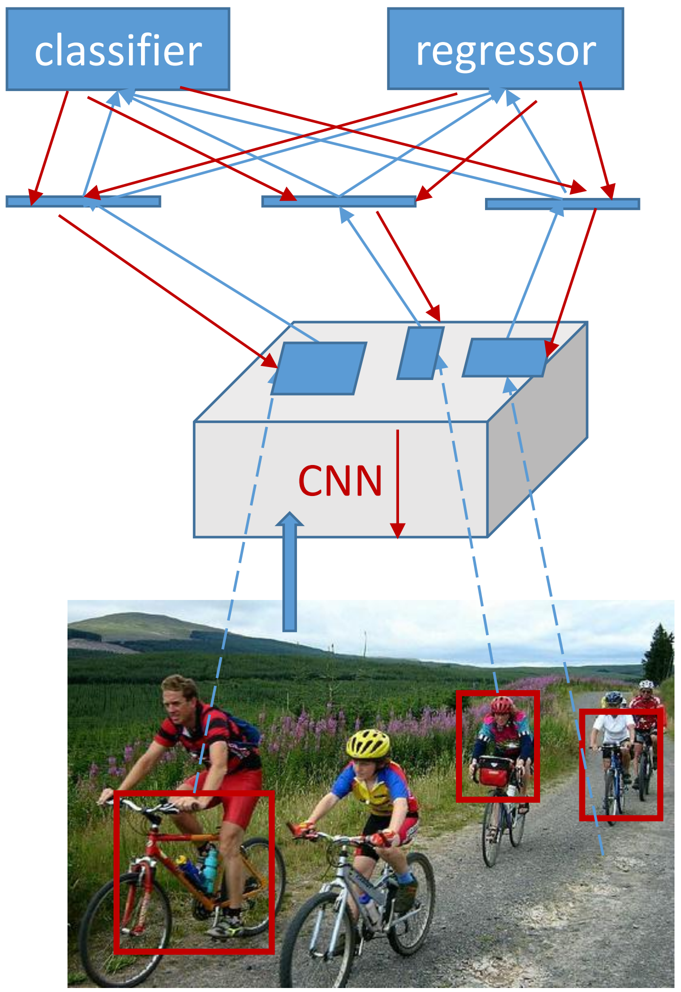
Fast R-CNN

- Generate region proposal
  - Candidate object regions
  - Using existing methods
- Extract CNN feature
  - For the **whole input image**
  - Forward-propagate the whole image via CNN
- Get CNN feature for each region
  - Using the region coordinates to locate areas in CNN feature maps
  - Extract the CNN feature from the areas
- Predict label for each region
  - Like image classification
  - Linear layer + softmax
- Regress bounding box for each region
  - Linear regression for each value
  - 4 values (coordinates, or x,y,h,w)



Fast RCNN

- Run CNN forwarding once
- End-to-end training
  - Softmax classifier
  - Linear regressor for bounding box



YOLO [3]

- Extract CNN feature for the whole image
- Divide the input image into fixed number of grids (e.g. 7x7)
  - For each cell
    - generate B (e.g. 3) bounding candidate boxes with different aspect ratios
    - Get the feature of each candidate box
    - Use Softmx to do label classification; linear regressor for coordinates refinement.
- NO Offline candidate region generation
- Notebook

Fast!
Not very accurate!

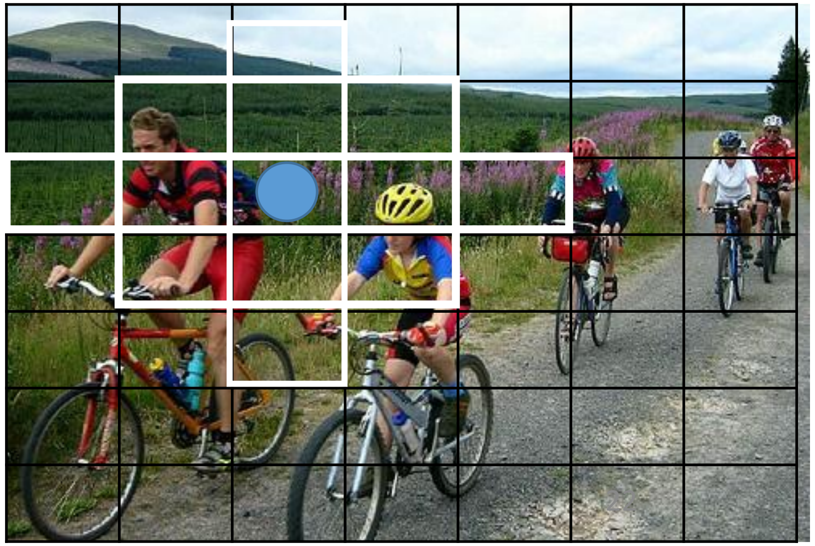


Image segmentation

- Label each pixel with a class
- Training label
  - A class (index) per pixel
- Prediction output
  - For each pixel, a probability vector (one per class)

Training label:
(0, 0) bicycle …
(200, 80) people (200, 81) people

Prediction output:
(0, 0) background 0.9; mountain: 0.1
…
(200, 80) people 0.8; bicycle 0.1; tree: 0.1

Image segmentation

- Applications
  - Medical image analysis
  - Self-driving car
- Evaluation [1]
  - Matched pixels = the predicted class of a pixel is the truth class
  - Mean IoU = average over all classes{#matched pixels/(truth pixels U predicted pixels)}

Image segmentation

- Solution
  - Encoder to extract a semantic-rich representation
    - For label prediction
    - Subsampling by (pooling or convolution with stride > 1)
  - Decoder to incorporate location information
    - To generate a final feature map as the same size as the input
    - Upsampling
  - Loss
    - Softmax loss for each pixel

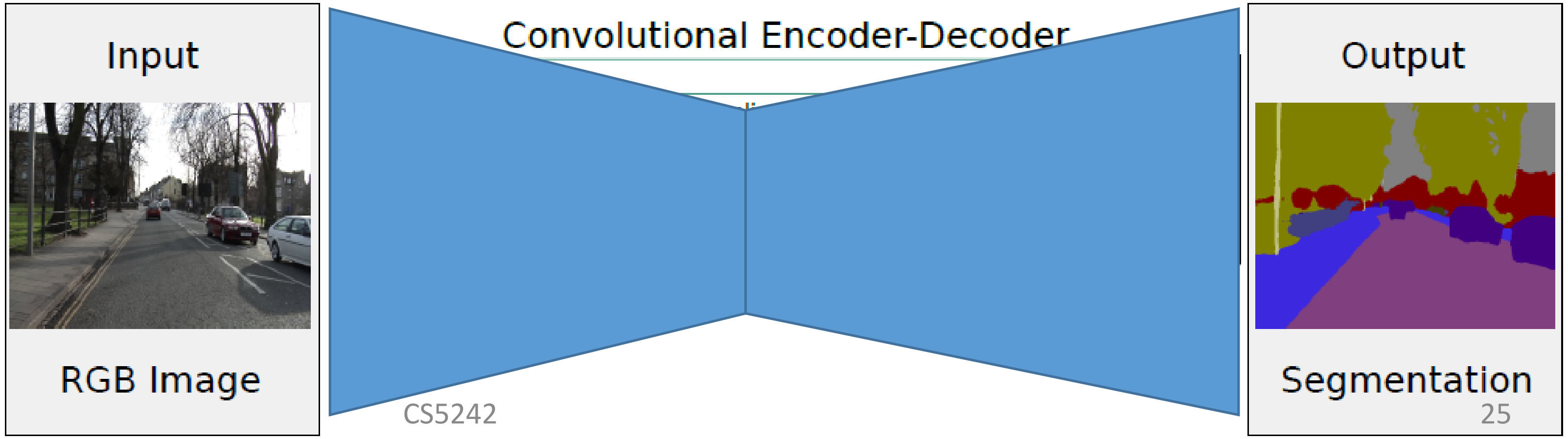


Image segmentation

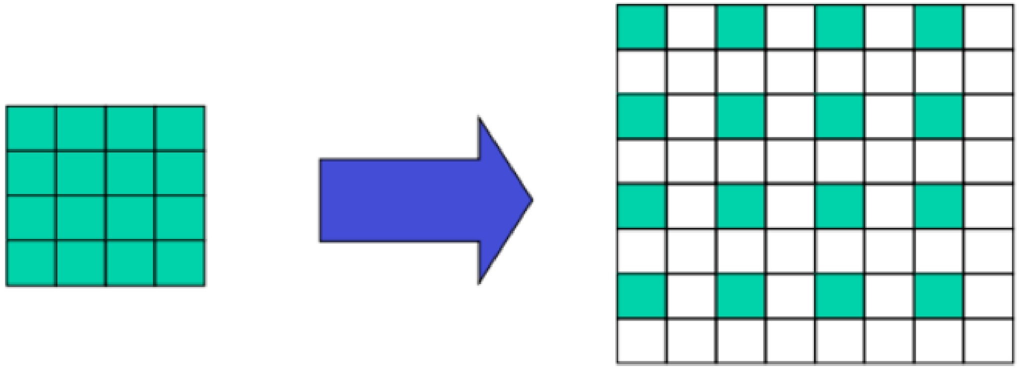
- Upsampling
  - Nearest neighbour

3	2
0	1

3	3	2	2
3	3	2	2
0	0	1	1
0	0	1	1

Image segmentation

- Bilinear upsampling



Original image:  x 10

Nearest-neighbor interpolation

Bilinear interpolation

- The empty pixels are initially set to 0
- Convolve with a (Gaussian, or another) filter
- If the filter sums to 1, multiply the result by 4
    - ¾ of the new image was initially 0

Image segmentation

- Max unpooling

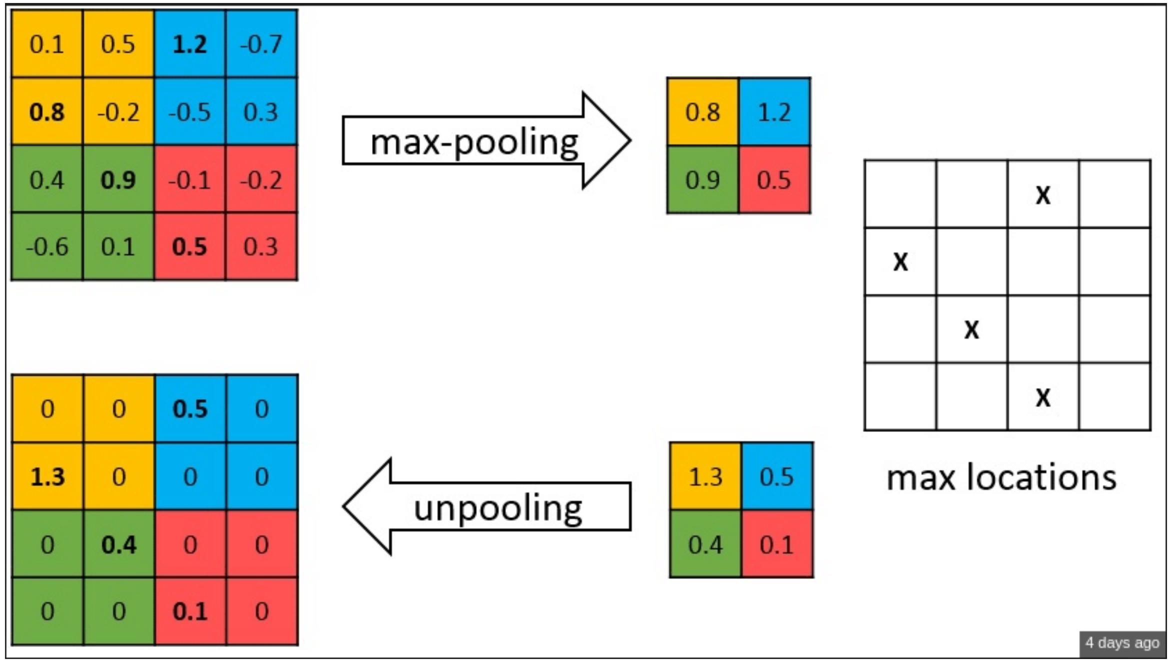
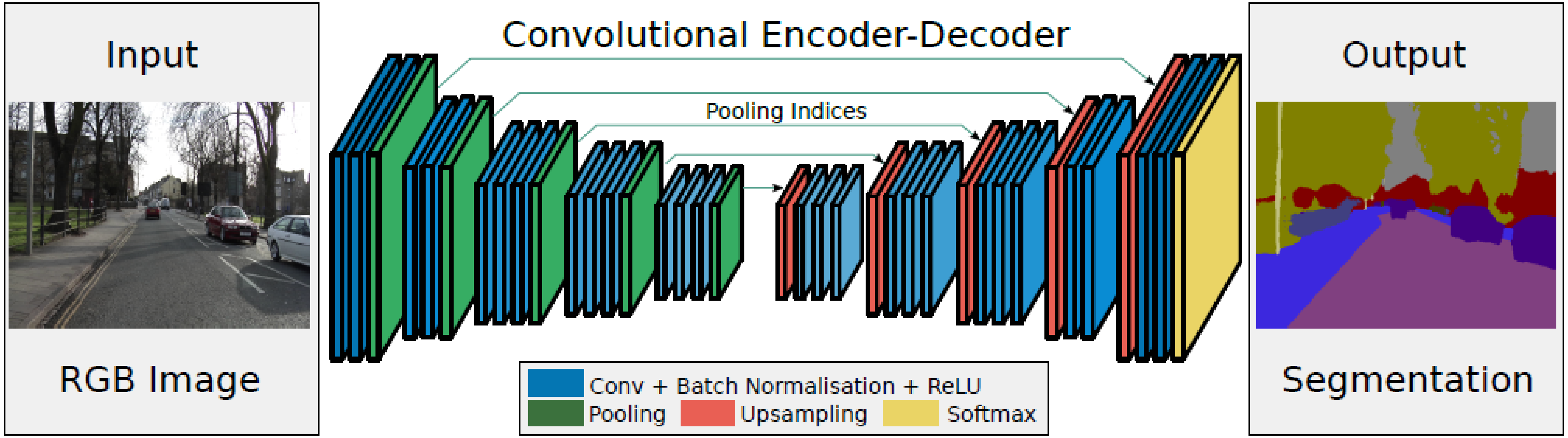


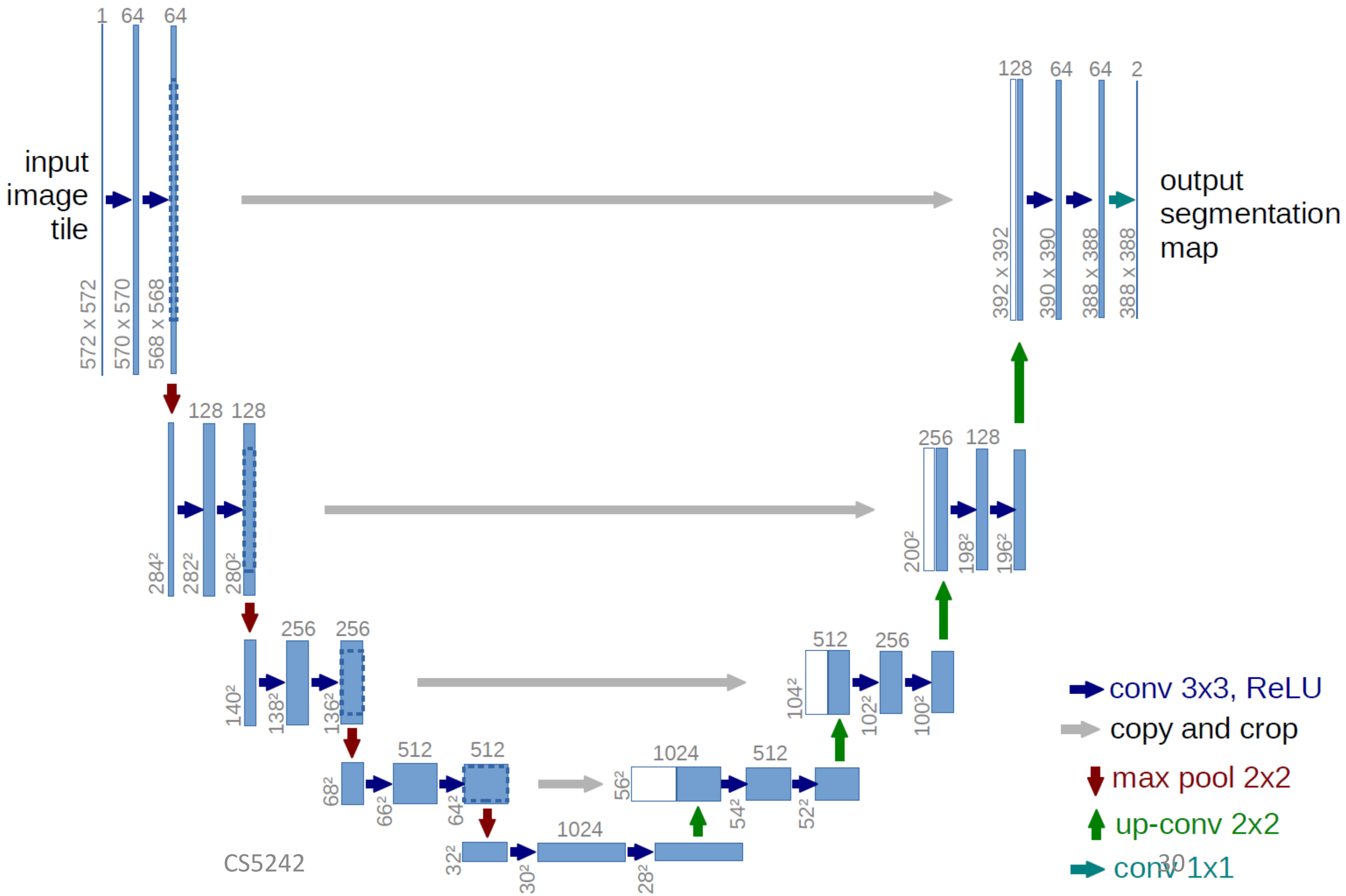
Image segmentation

- SegNet [4]
  - Max unpooling



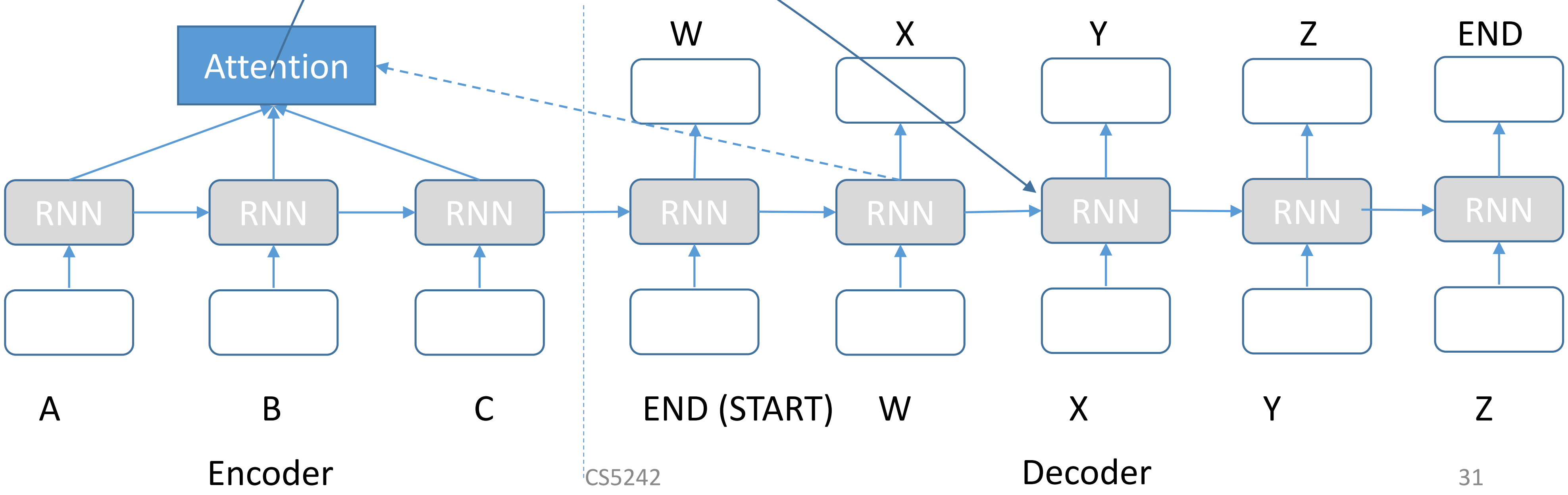
U-Net[5]

- Prototxt visualization
- Input and output
  - Different size
  - Due to valid padding
- Examples
- 1, 2



Attention modelling [6]

- Each output word depends on
  - All input words (hidden states), each with different contribution
    - Singapore MRT is not stable
    - 新加坡 地铁 不 稳定



Attention modelling [6]

- Encoder
  - Input a=[0.1,0,1], b=[1,0.1,0], c=[0.2,0.3,1]
  - Hidden representation (vector)
    - h1, h2, h3
    - h1=tanh(Ua+Wh0)
    - h2=tanh(Ub+Wh1)
    - h3=tanh(Uc+Wh2)

- Decoder
  - Hidden state s0 = [0,0,0] or h3
    - To compute the weights of h1, h2, h3 for computing s1
      - e_{11} = a(s0, h1), e_{12}=a(s0, h2), e_{13}=a(s0, h3)
      - a(s0, h1) = $v^T\tanh(W_a s_0 + U_a h_1)$
      - a(s0, h2) = $v^T\tanh(W_a s_0 + U_a h_2)$
      - a(s0, h3) = $v^T\tanh(W_a s_0 + U_a h_3)$
      - $k_{11} = \exp(e_{11}) / (\exp(e_{11})+\exp(e_{12})+\exp(e_{13}))$
      - $k_{12} = \exp(e_{12}) / (\exp(e_{11})+\exp(e_{12})+\exp(e_{13}))$
      - $k_{13} = \exp(e_{13}) / (\exp(e_{11})+\exp(e_{12})+\exp(e_{13}))$
      - c11=k_{11}h1+k_{12}h2+k_{13}h3
  - $s_t = (1 - z_t) \circ s_{t-1} + z_t \circ \tilde{s}_t$
  - $\tilde{s}_t = \tanh(\mathrm{W}([r_t \circ s_{t-1}, \mathrm{E}y_{t-1}, c_t])$
  - $r_t = \sigma(W_r[s_{t-1}, \mathrm{E}y_{t-1}, c_t])$
  - $z_t = \sigma(W_z[s_{t-1}, \mathrm{E}y_{t-1}, c_t])$

More Saturday sessions

- Other topics?
- Assignments answers will be uploaded

Reference

- [1]Yunchao Gong, Yangqing Jia, Thomas Leung, Alexander Toshev, Sergey Ioffe. Deep convolutional ranking for multilabel image annotation. https://arxiv.org/pdf/1312.4894.pdf
- [2]Ross Girshick. Fast R-CNN. https://arxiv.org/abs/1504.08083
- [3]Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection. https://arxiv.org/abs/1506.02640
- [4]SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. Vijay Badrinarayanan, Alex Kendall, Roberto Cipolla. 2016
- [5]U-Net: Convolutional Networks for Biomedical Image Segmentation. Olaf Ronneberger, Philipp Fischer, Thomas Brox. 2015
- [6] https://machinelearningmastery.com/how-does-attention-work-in-encoder-decoder-recurrent-neural-networks/
- [7] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in IEEE Conference on Computer Vision and Pattern
- Recognition (CVPR), 2014