Advanced Topics in CNN & RNN

LEADINGINDIA.AI

NATIONWIDE AI SKILLING & RESEARCH INITIATIVE

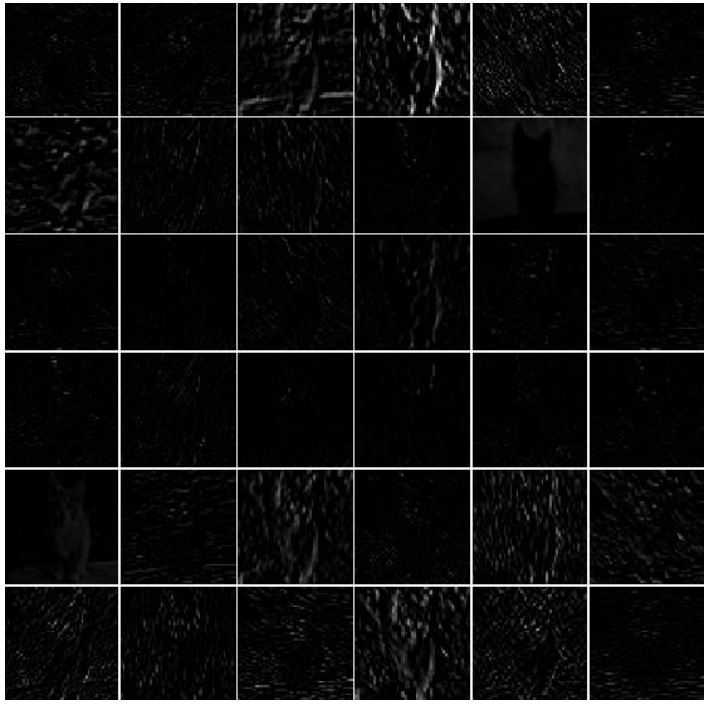
Session Coverage

- CNN Visualization
- New Generation Convolutions
- Advanced Object Detection
- Image Segmentation
- Style Transfer
- Attention Model
- Word Embeddings

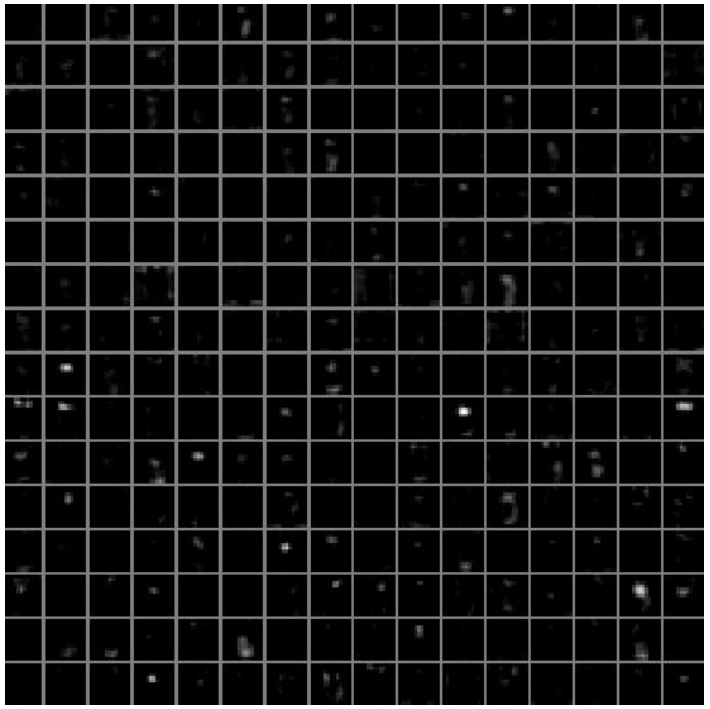
Layer Visualization using Activations

AlexNet Layer Activations

Input Image



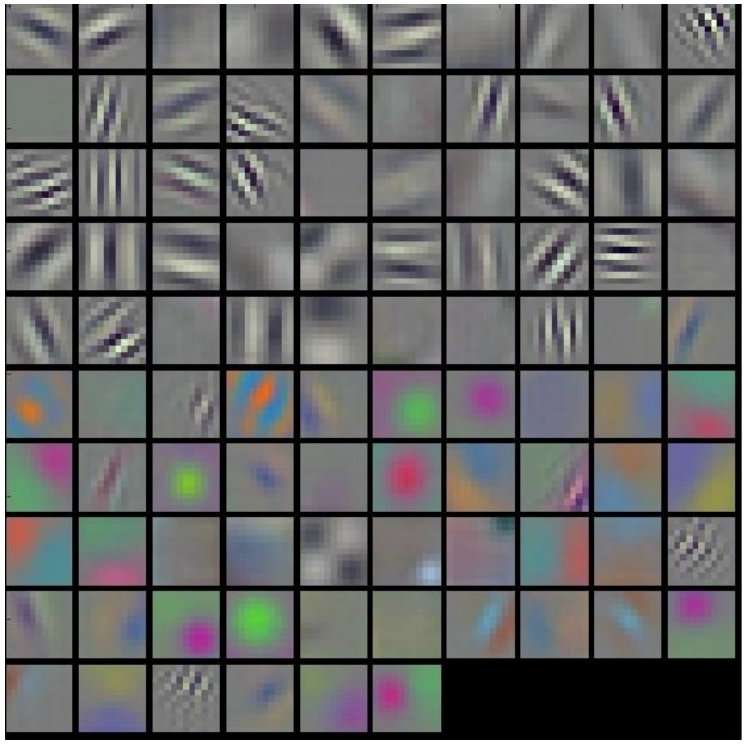
Activations on the 1st CONV layer



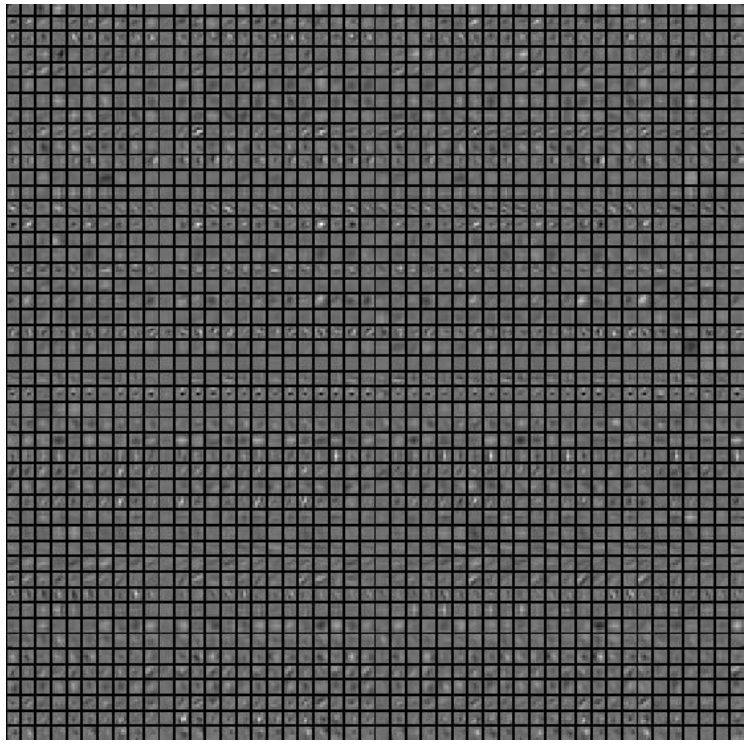
Activations on the 5th CONV layer

Visualizing the Filter Weights

Weights of 1st CONV layer

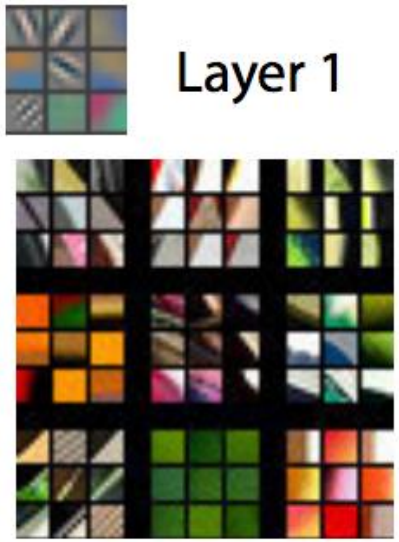


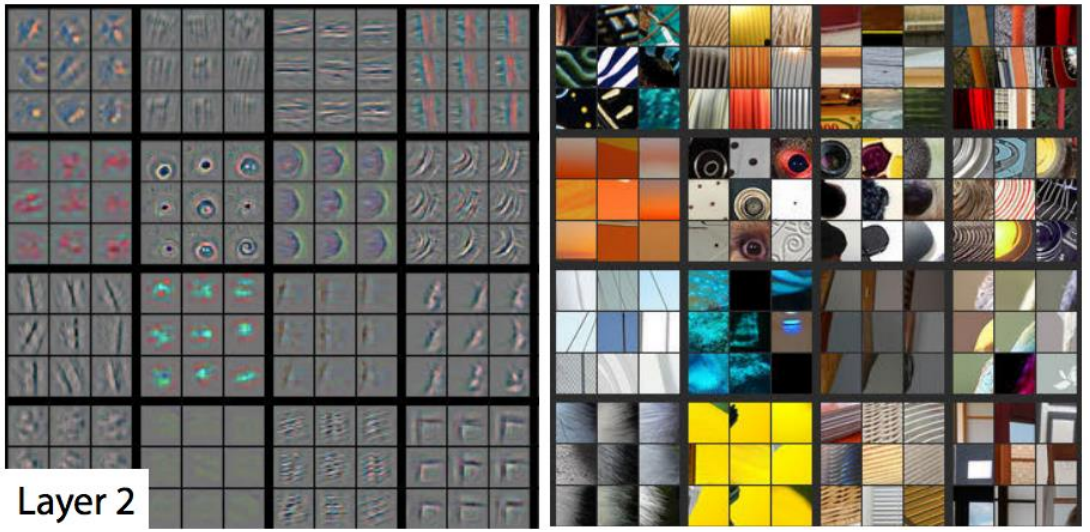
Weights of 2nd CONV layer

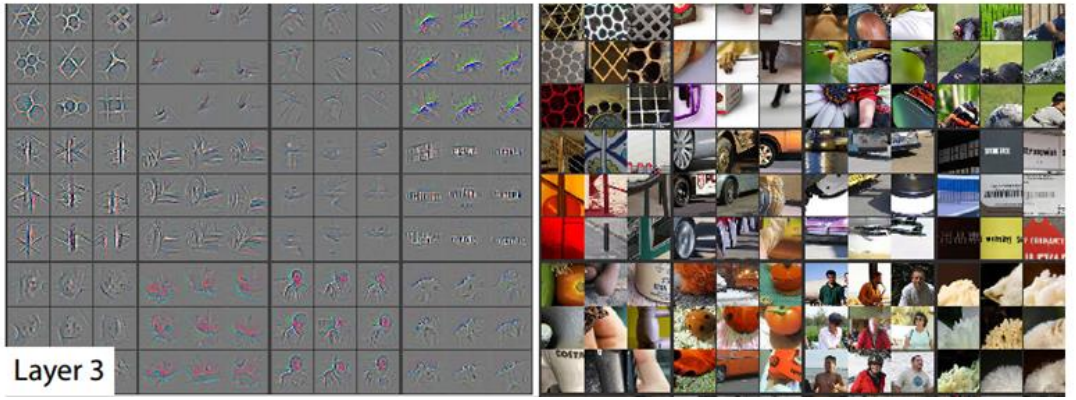


Notice that the first-layer weights are very nice and smooth, indicating nicely converged network. The 2nd CONV layer weights are not as interpretable, but it is apparent that they are still smooth, well-formed, and absent of noisy patterns.

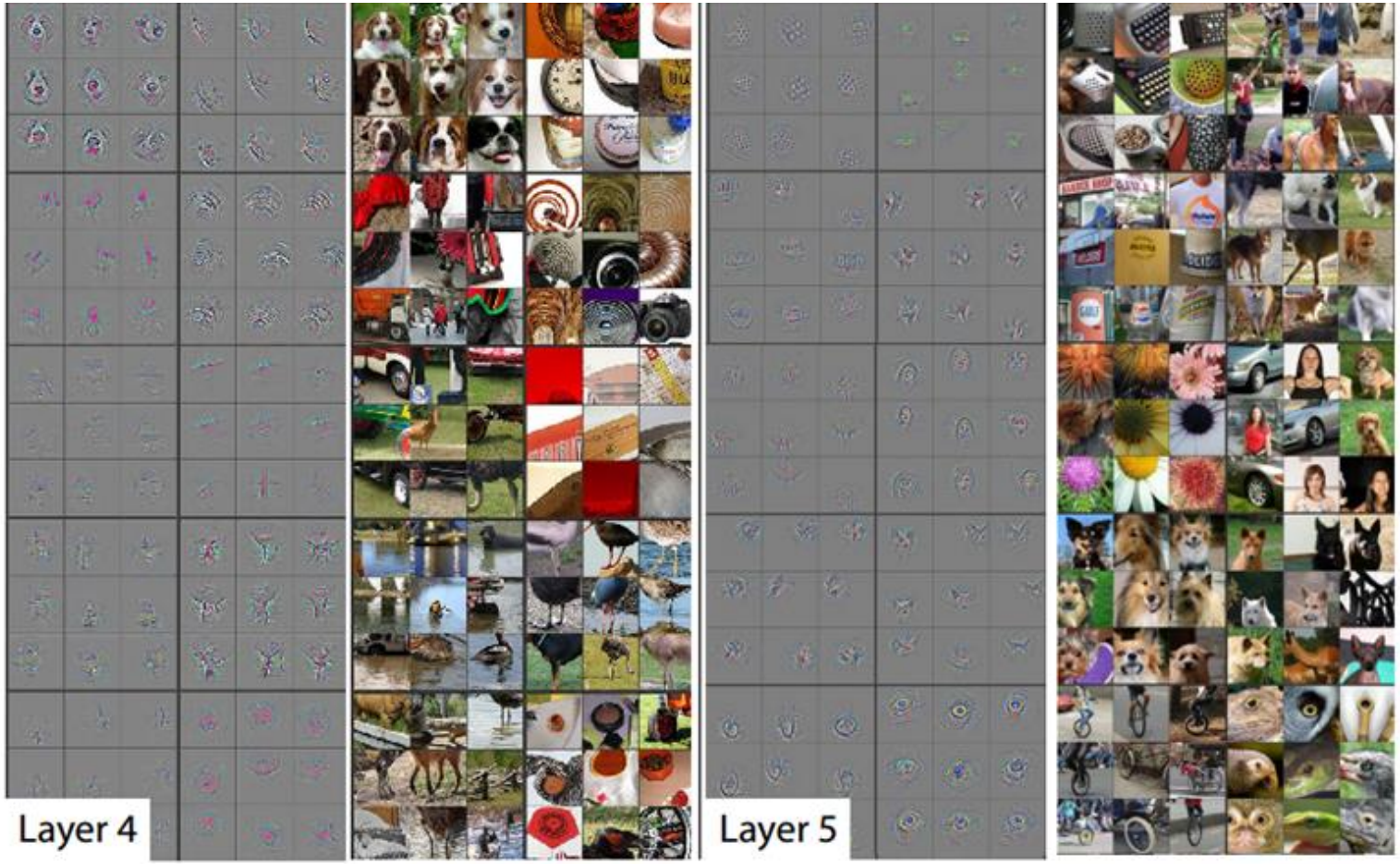
Deconv Networks







Deconv Networks



Retrieving images that maximally activate a neuron

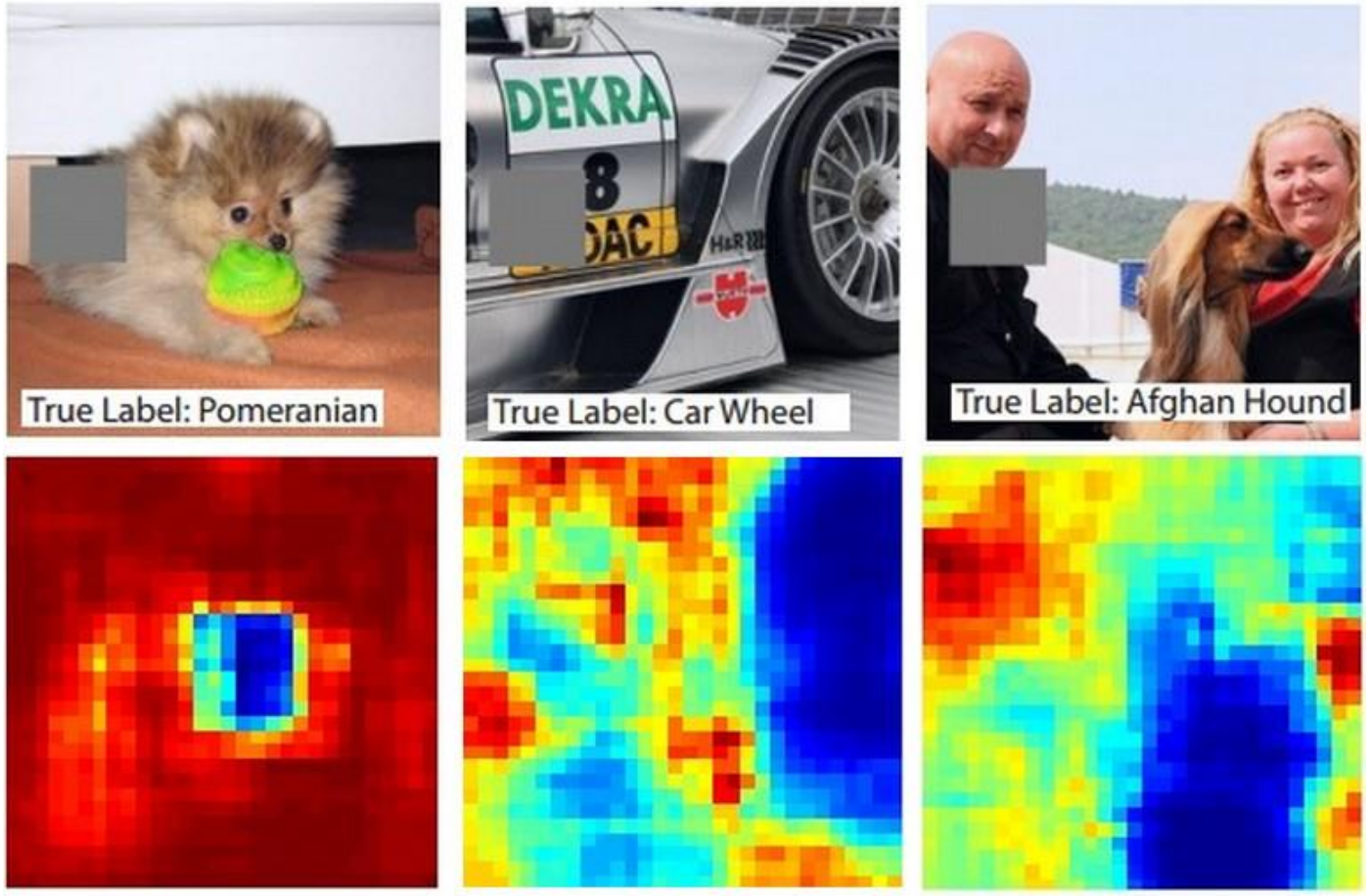
Maximally activating images for some POOL5 (5th pool layer) neurons of an AlexNet

Embedding the codes with t-SNE

t-SNE embedding of a set of images based on their CNN codes. Images that are nearby each other are also close in the CNN representation space, which implies that the CNN "sees" them as being very similar. Notice that the similarities are more often class-based and semantic rather than pixel and color-based.

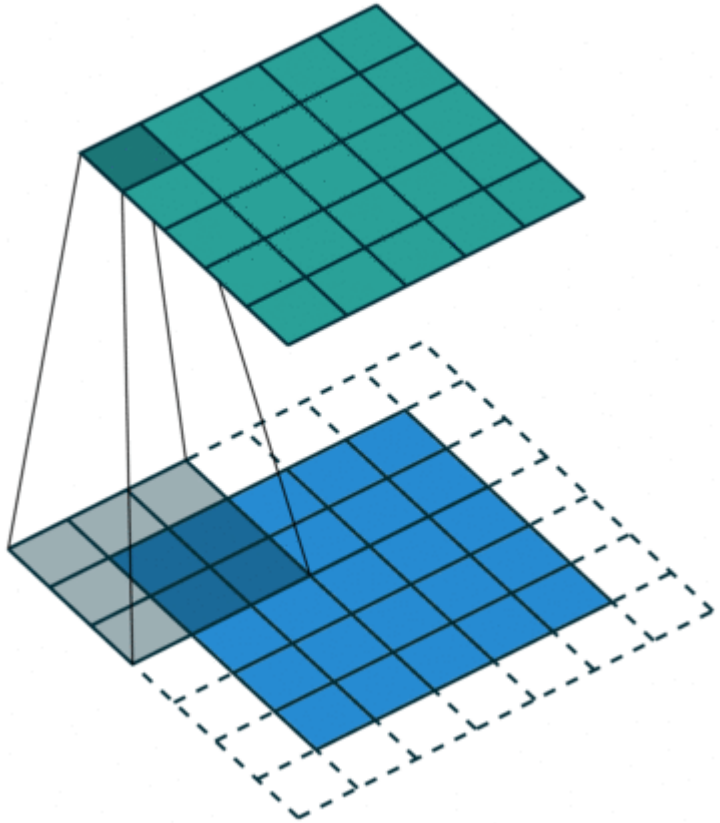
Occluding Parts of the Image

Three input images (top). Notice that the occluder region is shown in grey. As we slide the occluder over the image we record the probability of the correct class and then visualize it as a heatmap (shown below each image).



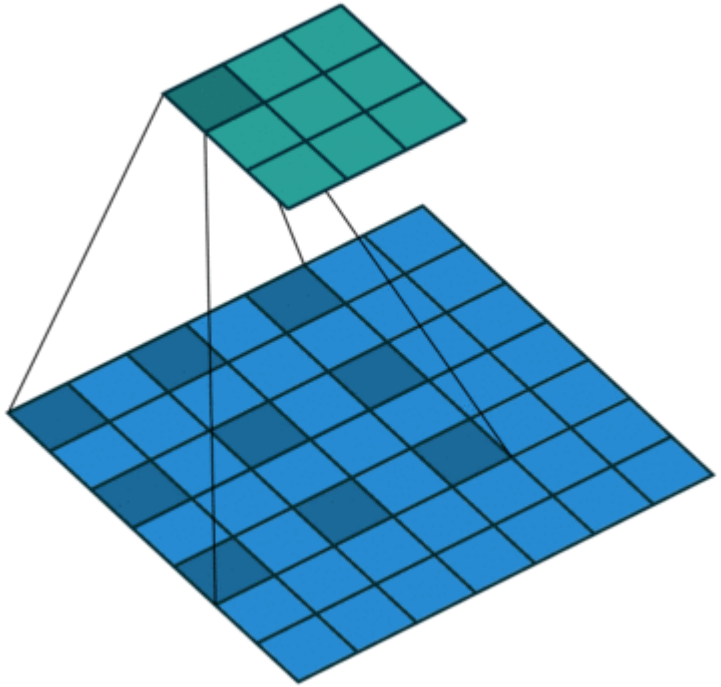
2D Convolution

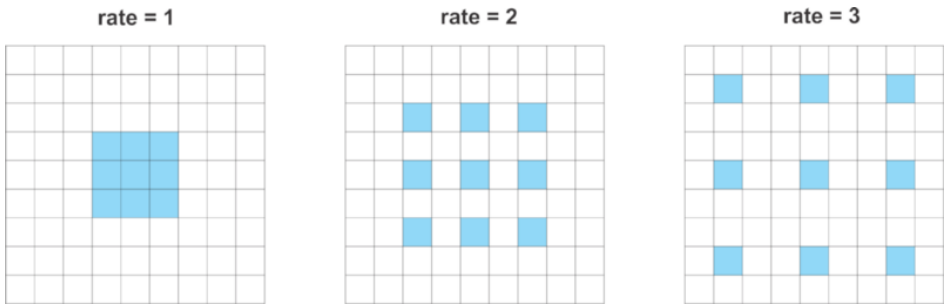
•**Kernel Size**: The kernel size defines the field of view of the convolution. A common choice for 2D is 3 i.e 3x3 pixels.
•**Stride**: The stride defines the step size of the kernel when traversing the image. While its default is usually 1, we can use a stride of 2 for downsampling an image similar to MaxPooling.
•**Padding**: The padding defines how the border of a sample is handled. A (half) padded convolution will keep the spatial output dimensions equal to the input, whereas unpadded convolutions will crop away some of the borders if the kernel is larger than 1.



Atrous/Dilated Convolution

- Dilated convolutions are particularly popular in the field of real-time segmentation.
- Dilated convolutions introduce another parameter to convolutional layers called the **dilation rate,** that defines a spacing between the values in a kernel.
- A 3x3 kernel with a dilation rate of 2 will have the same field of view as a 5x5 kernel, while only using 9 parameters. Imagine taking a 5x5 kernel and deleting every second column and row.
- Use them if you need a wide field of view and cannot afford multiple convolutions or larger kernels.





Dilated Convolution

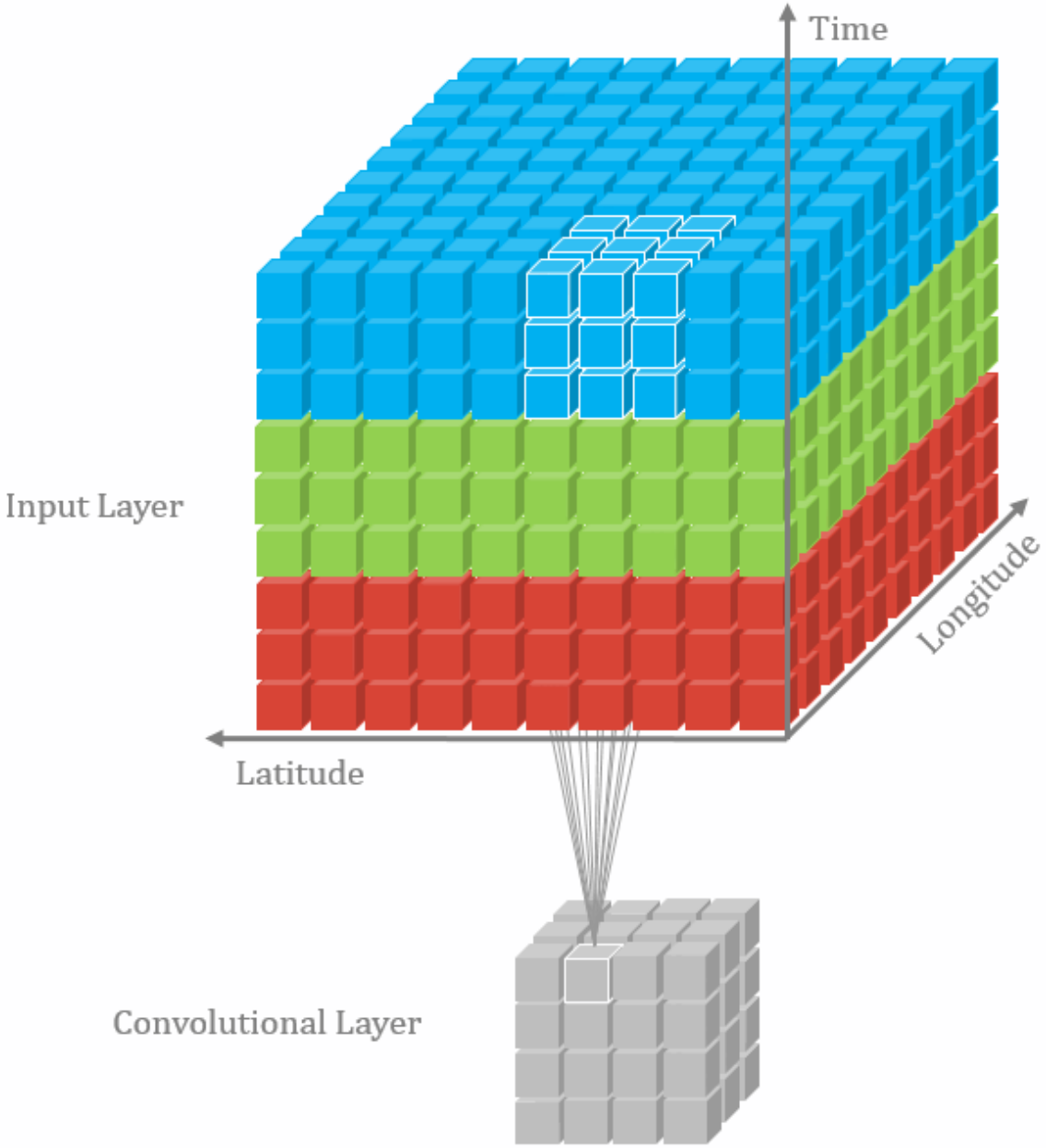
- Dilated convolutions have generally improved performance in semantic segmentation results
- The architecture is based on the fact that dilated convolutions support exponential expansion of the receptive field without loss of resolution or coverage.
- Allows one to have larger receptive field with same computation and memory costs while also preserving resolution.
- Pooling and Strided Convolutions are similar concepts but both reduce the resolution.

3D CNN

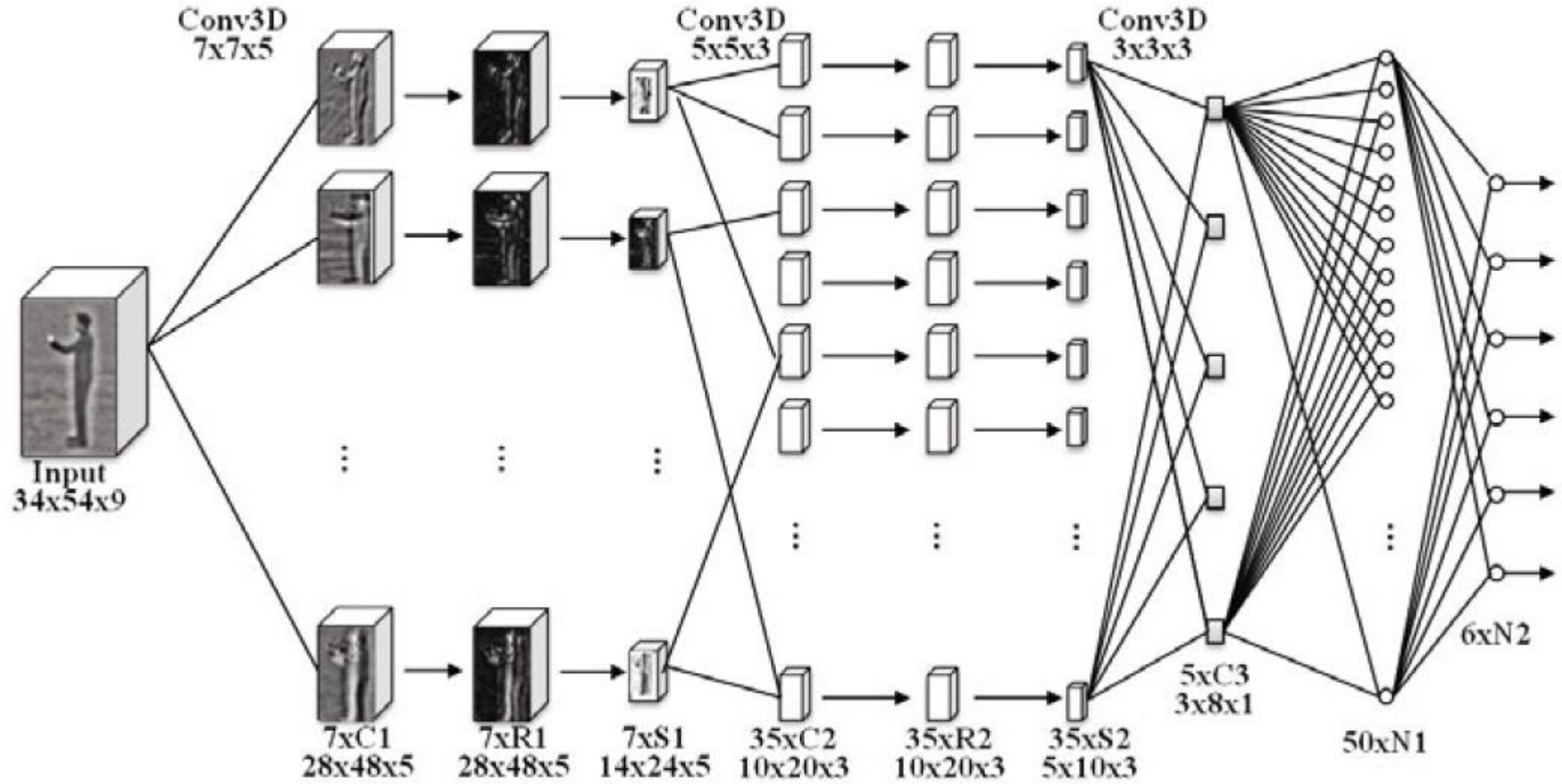
- 3D imageries such as MRI Scan, Human action in video sequence should be processed frame-by-frame
- Temporal property within the 3D imagery is important in extracting the features
- 2D convolution extracts features from only spatial dimensions
- 3D convolution operates on the input volume not only in the *X* and *Y* dimensions but also in *Z* dimension
- Building a 3D CNN requires, 3D Convolution as well as 3D pooling

2D vs 3D Convolution

1	6	5
7	10	9
7	10	8

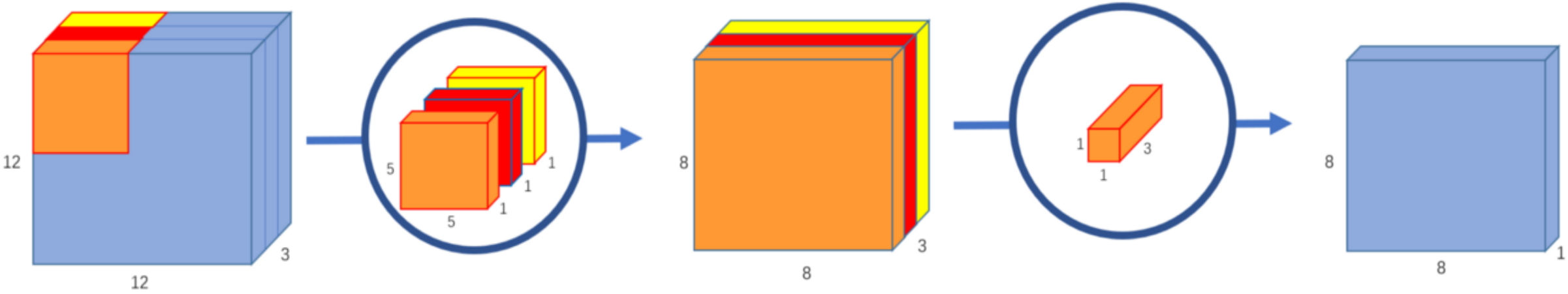


3D CNN for Action Recognition



Depthwise Separable Convolutions

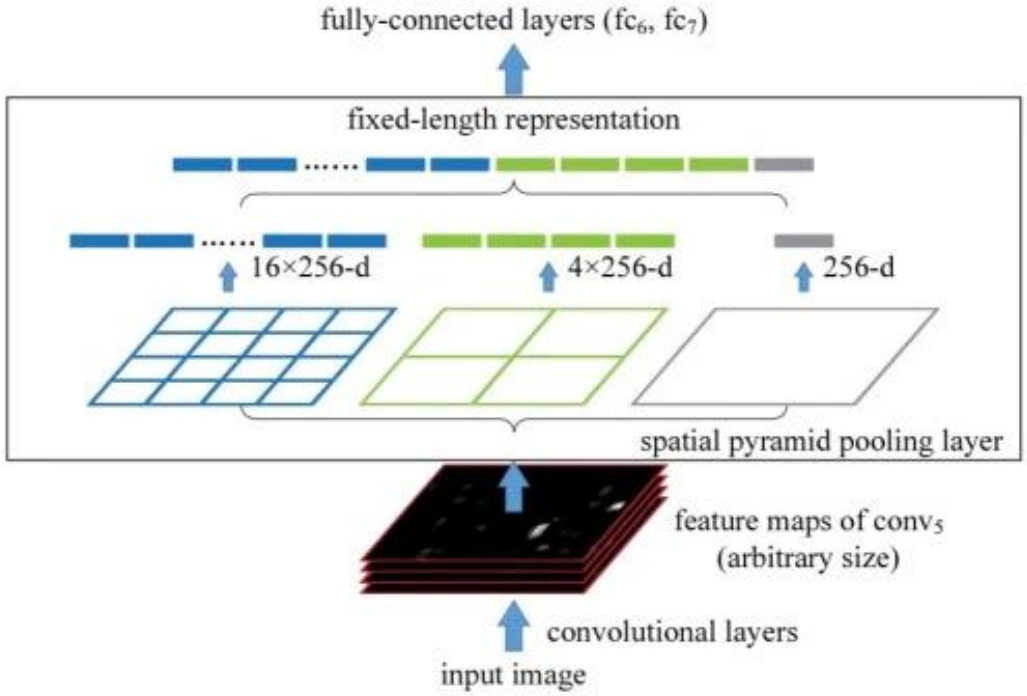
- Depthwise convolutions is a technique for performing convolutions with less number of computations than a standard convolution operation. This involves breaking down the convolution operation into two steps:
  - Depthwise convolution
  - Pointwise convolution



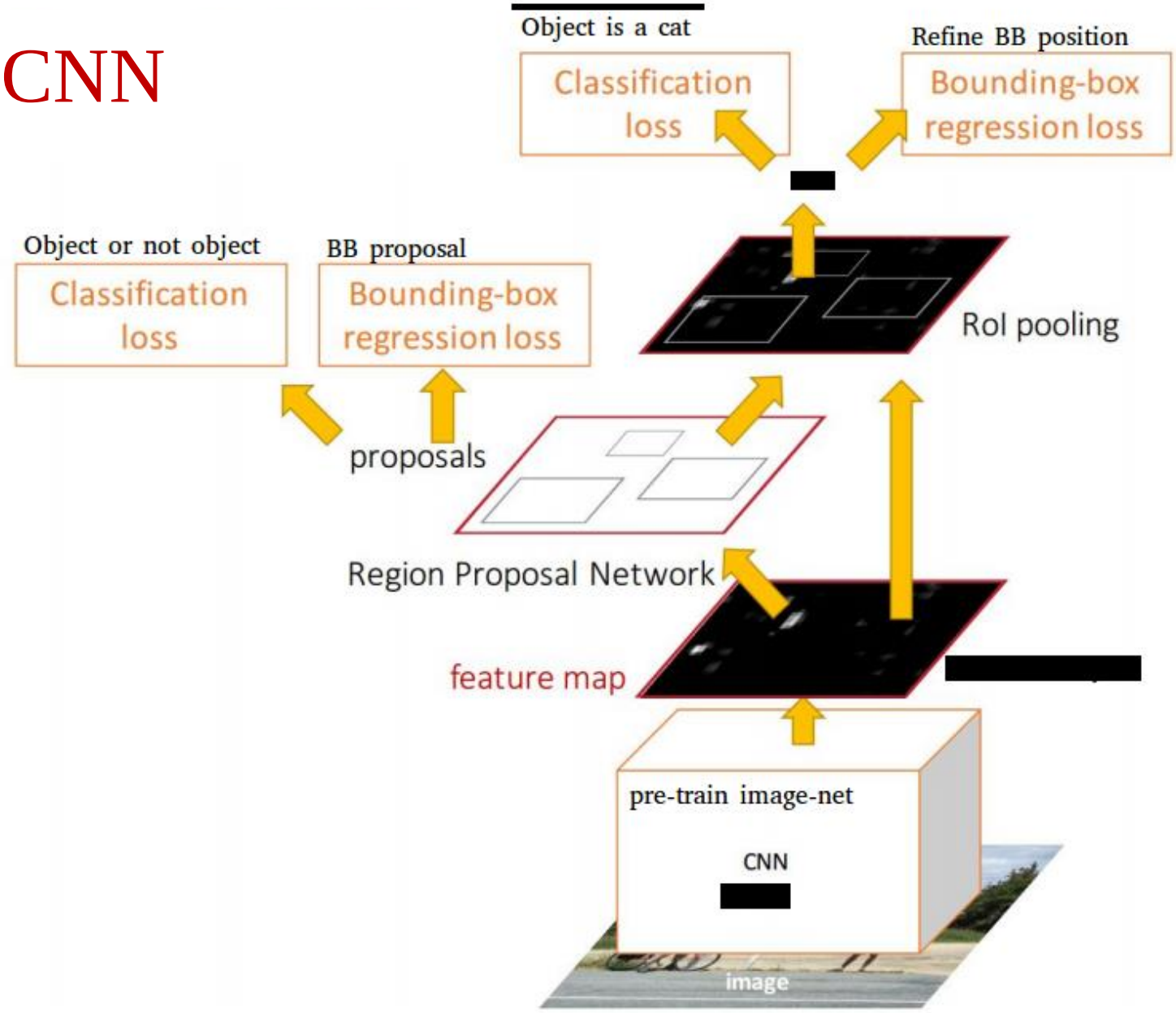
We can use 256, 1x1x3 filters over the input of 8x8x3 and get an output shape of 8x8x256

Spatial Pyramid Pooling Layer

- **Spatial Pyramid Pooling** (SPP) allows CNN to use input images of any size, not only 224*224
- Convolutional layers operate on any size, but fully connected layers need fixed-size inputs
- Add a new SPP layer on top of the last convolutional layer, before the fully connected layer
- The SPP layer operates on each feature map independently.
- The output of the SPP layer is of dimension k*M, where k is the number of feature maps the SPP layer got as input and M is the number of bins
- Highly used in image segmentation (such as **DeepLab**) where image dimension has to be preserved

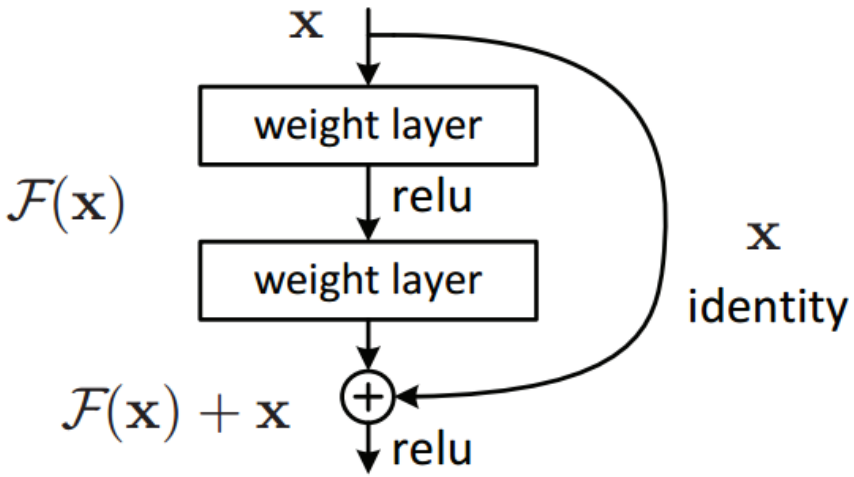


Faster R-CNN

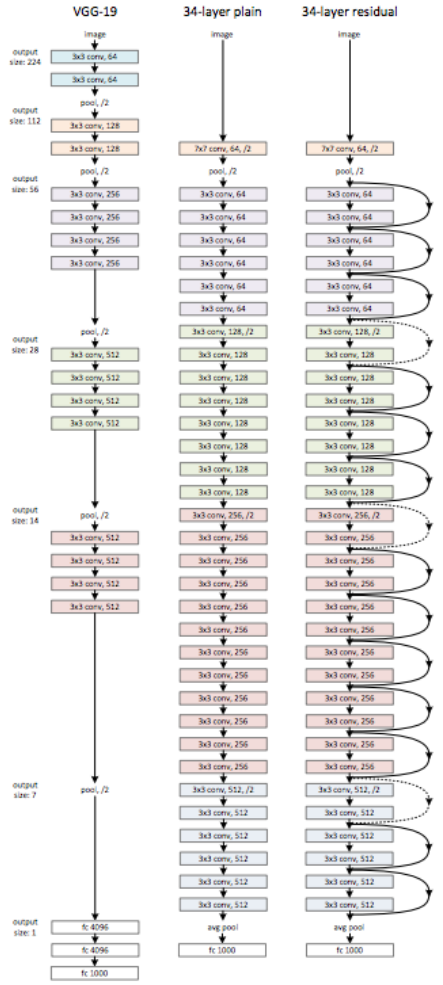

Object is a cat
Classification loss
Refine BB position
Bounding-box regression loss
Object or not object
Classification loss
BB proposal
Bounding-box regression loss
RoI pooling
proposals
Region Proposal Network
feature map
pre-train image-net
CNN
image

Residual Block

- Deep Networks suffer from Vanishing Gradients
- Residual block allows us to add input with the output of successive layers to maintain the activation till the end
- **Constraints:**
  - In case of dense layers, number of nodes has to be same for i/p and o/p
  - In case of convolution, filters should be same for i/p and o/p. Padding is mandatory. Pooling should be done before applying the residual block
- Applied to all other domains of deep learning including speech and natural language processing



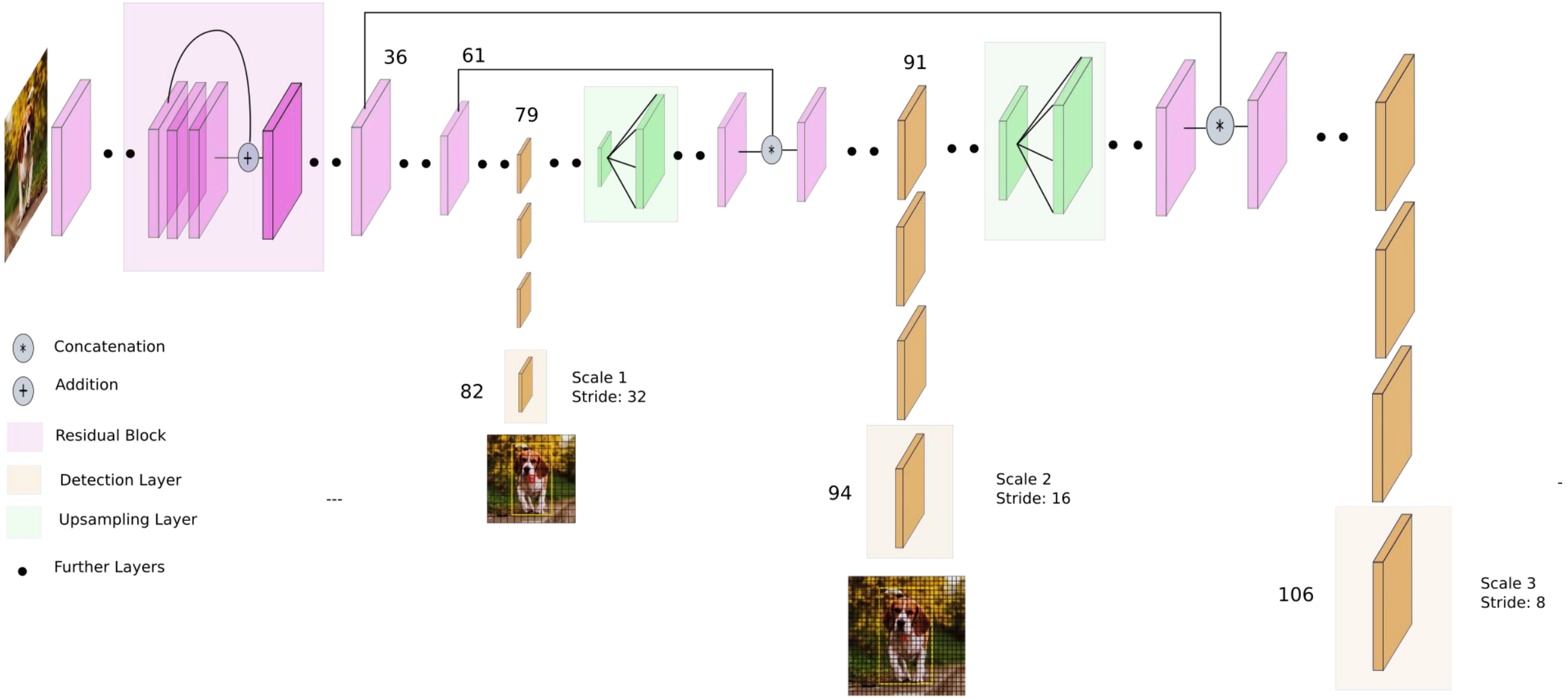
Residual Block



VGG16 vs ResNet

YOLO v3

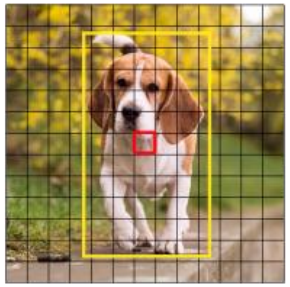
- YOLO – You Only Look Once
- Labelled as “YOLO v3: Better, not Faster, Stronger”
- YOLO v3 was aimed at improving the accuracy i.e. little slower
- Previous version ran on 45 FPS on a Titan X, v3 clocks about 30 FPS
- YOLO v3 is based on **106 layer** fully convolutional underlying architecture
- **Detections at different layers** helps address the issue of detecting small objects, a frequent complaint with YOLO v2.



YOLO v3 network Architecture

YOLO v3

Prediction Feature Maps at different Scales

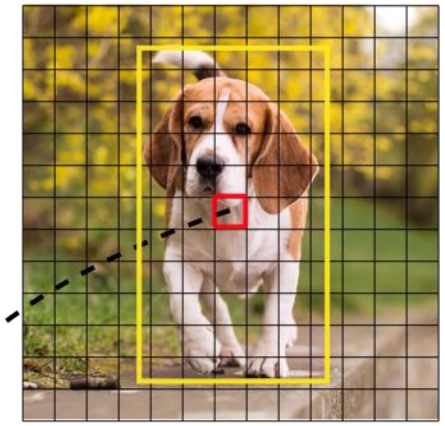


13 x 13

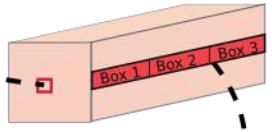
26 x 26

52 x 52

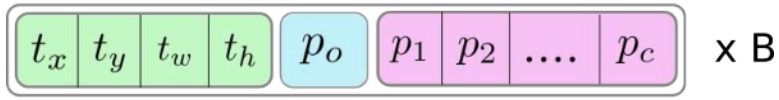
Image Grid. The Red Grid is responsible for detecting the dog



Prediction Feature Map



Attributes of a bounding box



Multiple Grids may detect the same object
NMS is used to remove multiple detections

What's new in YOLO v3

- **Logistic regression for confidence scores**: YOLOv3 predicts an confidence score for each bounding box using *logistic regression* while YOLO and YOLOv2 uses sum of squared errors for classification terms
- **No more softmax for class prediction:** When predicting class confidence, YOLOv3 uses multiple independent logistic classifier for each class rather than one softmax layer. Considering that one image might have multiple labels.
- **Darknet + ResNet as the base model:** The new Darknet-53 still relies on successive 3x3 and 1x1 conv layers, just like the original dark net architecture, but has residual blocks added.
- **Multi-scale prediction**: Inspired by image pyramid, YOLOv3 adds several conv layers after the base feature extractor model and makes prediction at three different scales among these conv layers. In this way, it has to deal with many more bounding box candidates of various sizes overall.
- **Skip-layer concatenation:** YOLOv3 also adds cross-layer connections between two prediction layers (except for the output layer) and earlier finer-grained feature maps. The model first up-samples the coarse feature maps and then merges it with the previous features by concatenation. The combination with finer-grained information makes it better at detecting small objects.

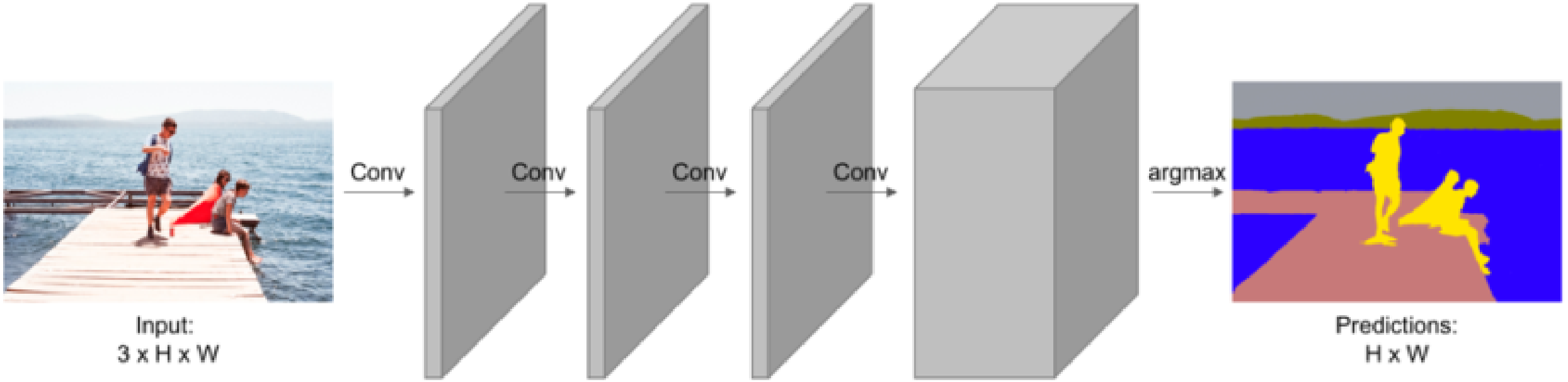
Semantic Segmentation

- *Semantic segmentation* describes the process of associating each pixel of an image with a class label such as *person*, *road*, *sky*, or *car*
- Traditional approaches involves unsupervised methods such as clustering, graph segmentation, etc.
- Unsupervised segmention is faster but fails to aggregate high-level visual features.
- Applications for semantic segmentation include:
  - Autonomous driving
  - Industrial inspection
  - Classification of terrain visible in satellite imagery
  - Medical imaging analysis

Fully Convolutional Network (FCN)

- Standard CNN architecture can be used for FCN by removing the fully connected layers
- Any standard CNN architecture can be used for FCN by **removing the fully connected layers**. The fully connected layers are replaced by a convolution layer.
- The depth is higher in the final layers and the size is smaller. Hence, 1D convolution can be performed to reach the desired number of labels.
- But for segmentation, the spatial dimension has to be preserved. Hence, the full convolution network is constructed **without a max pooling**

Fully Convolutional Network (FCN)

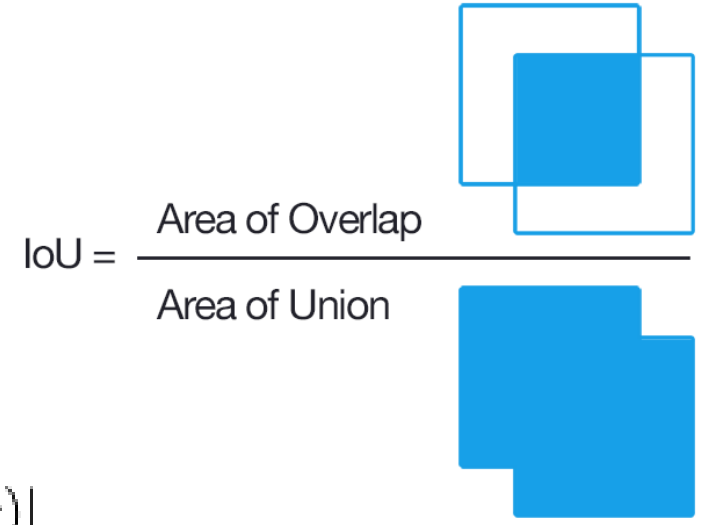


- The final layer has a depth equal to the number of classes. FCN is similar to object detection except that the spatial dimension is preserved.
- The output produced by the architecture will be coarse as some pixels may be mis-predicted, while the computation is high.

Segmentation Evaluation

- **Loss** for this network is computed by averaging the cross-entropy loss of every pixel and mini-batch
- **Accuracy**

$$acc(P, GT) = \frac{|\text{pixels correctly predicted}|}{|\text{total nb of pixels}|}$$



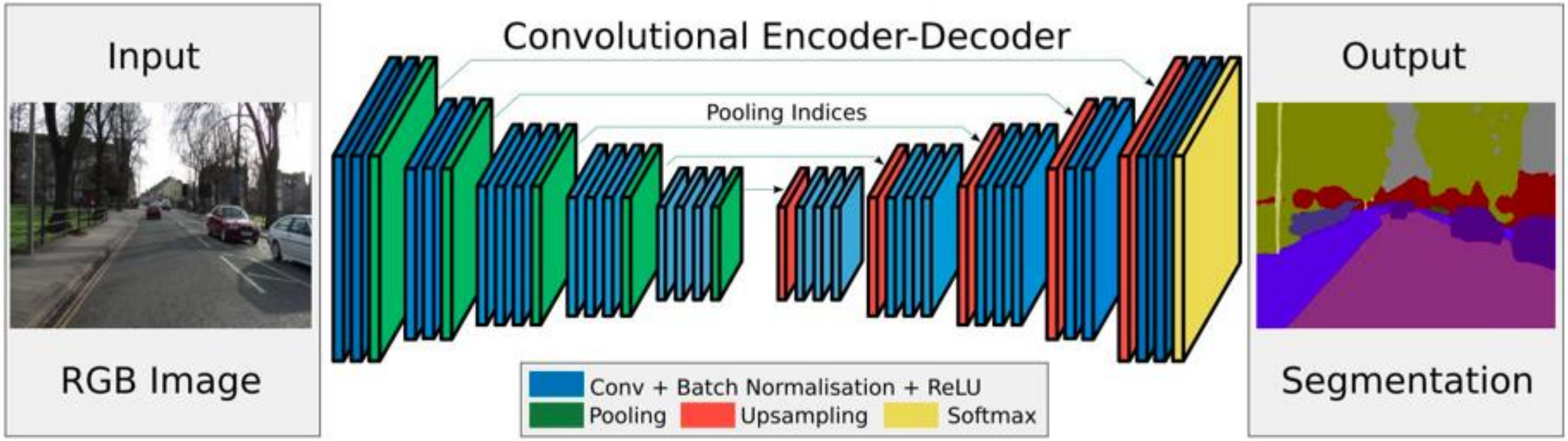
- **Jaccard (Intersection over Union)**

$$jacc(P(class), GT(class)) = \frac{|P(class) \cap GT(class)|}{|P(class) \cup GT(class)|}$$

SegNet

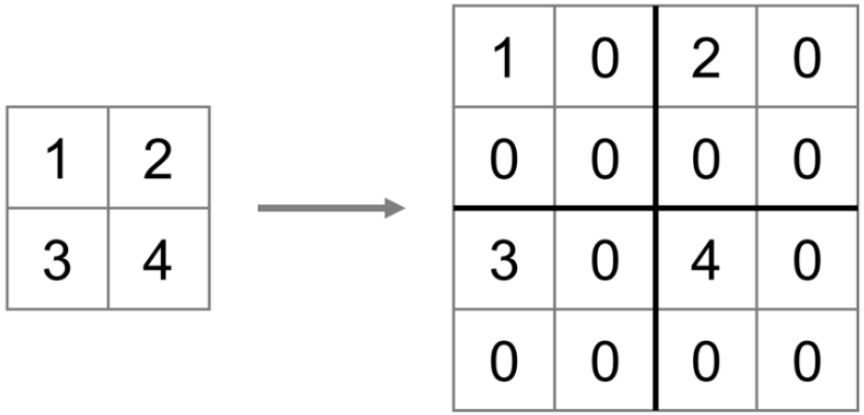
- **SegNet** has an encoder and decoder approach
- The encoder has various convolution layers and the decoder has various deconvolution layers.
- SegNet improves the coarse outputs produced by FCN. Because of this, it is less memory-intensive. When the features are reduced in dimensions, it is upsampled again to the image size by deconvolution, reversing the convolution effects.
- Deconvolution learns the parameters for upsampling. The output of such architecture will be coarse due to the loss of information in pooling layers

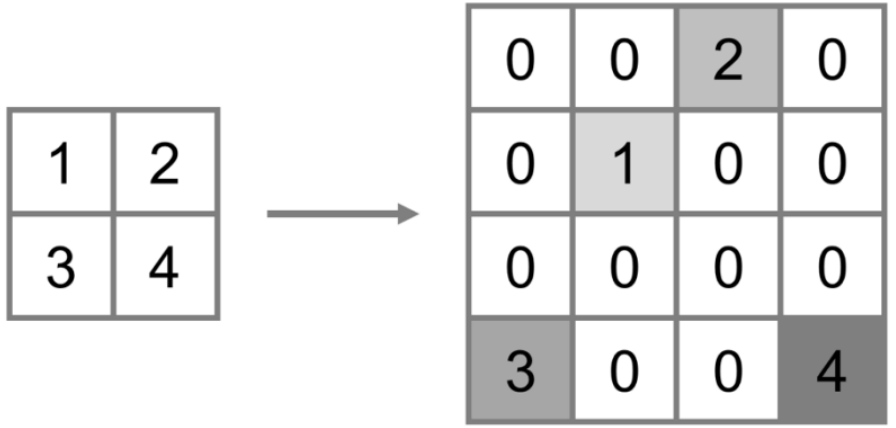
SegNet


Input
RGB Image
Convolutional Encoder-Decoder
Pooling Indices
Conv + Batch Normalisation + ReLU
Pooling
Upsampling
Softmax
Output
Segmentation

Upsampling the layers by pooling

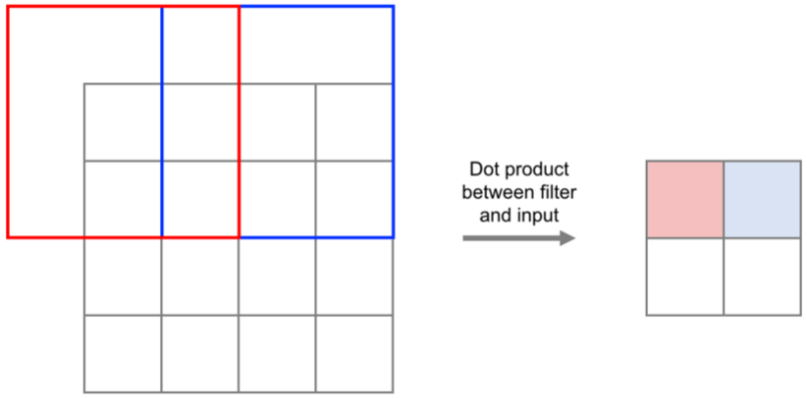
- Max pooling is a sampling strategy that picks the maximum value from a window. This could be reversed for upsampling. Each value can be surrounded with zeroes to upsample the layer
- The zeroes are added at the same locations, which are the numbers that are upsampled. Un-pooling can be improved by remembering the locations of downsampling and using it for upsampling





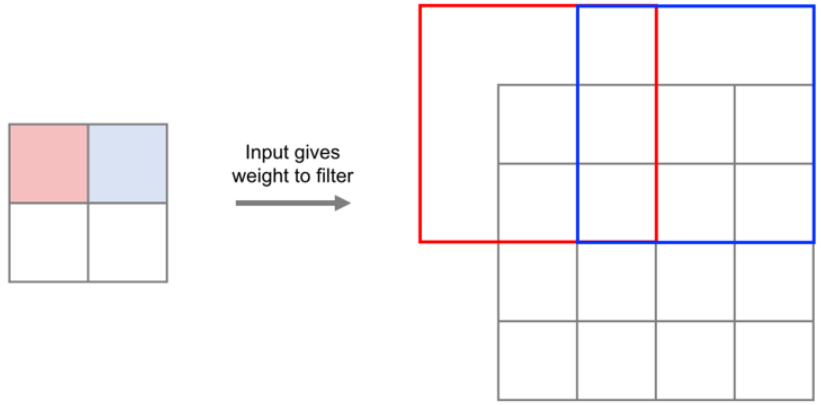
Sampling the layers by convolution

- The layers can be upsampled or downsampled directly using convolution. The stride used for convolution can be increased to cause downsampling. Downsampling by convolution is called **atrous convolution** or **dilated convolution** or **strided convolution**



Convolution

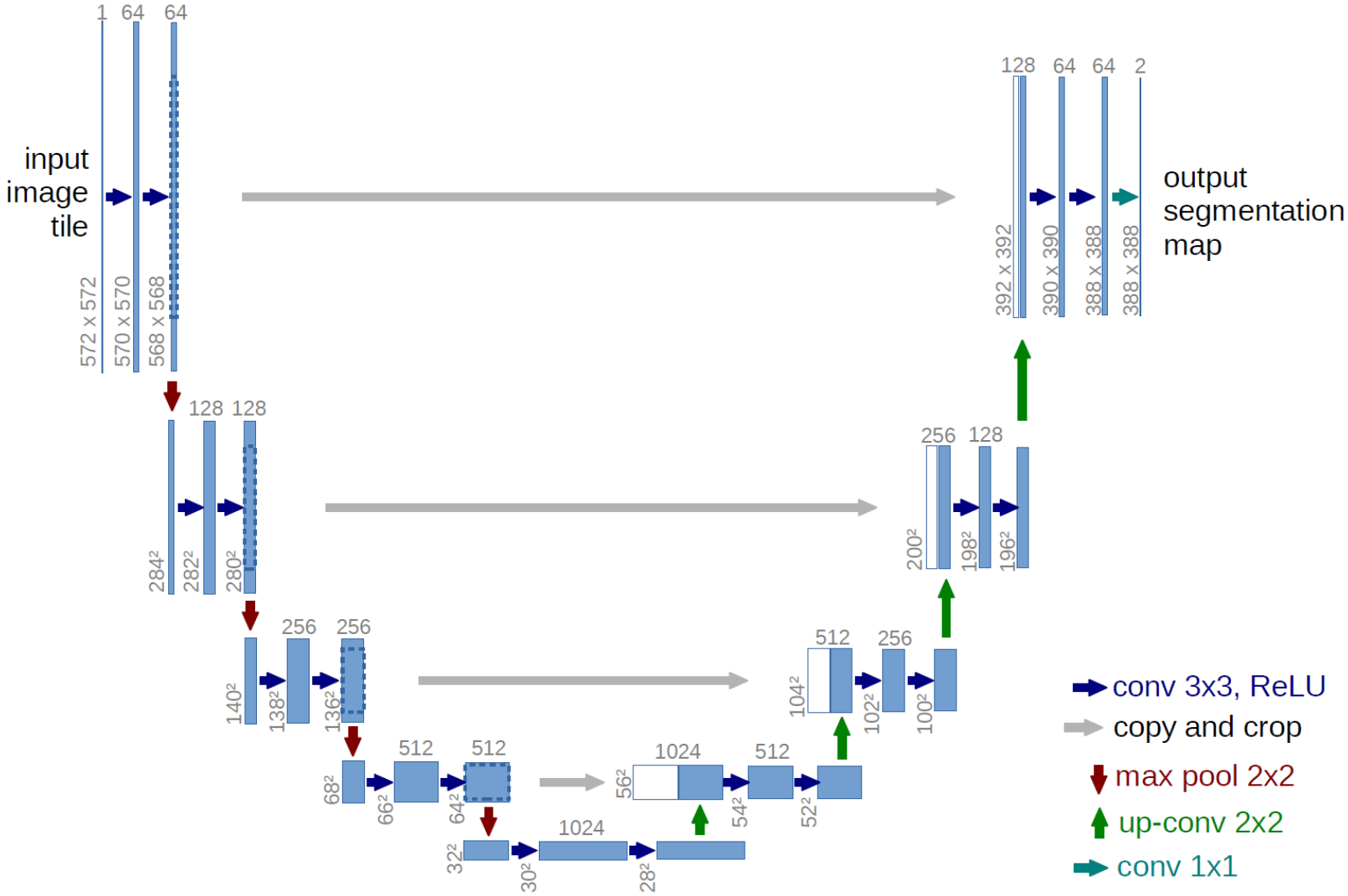
- Upsampling directly using a convolution can be termed as transposed convolution. Some other synonyms are **deconvolution**, **fractionally strided convolution**, or **up-convolution**



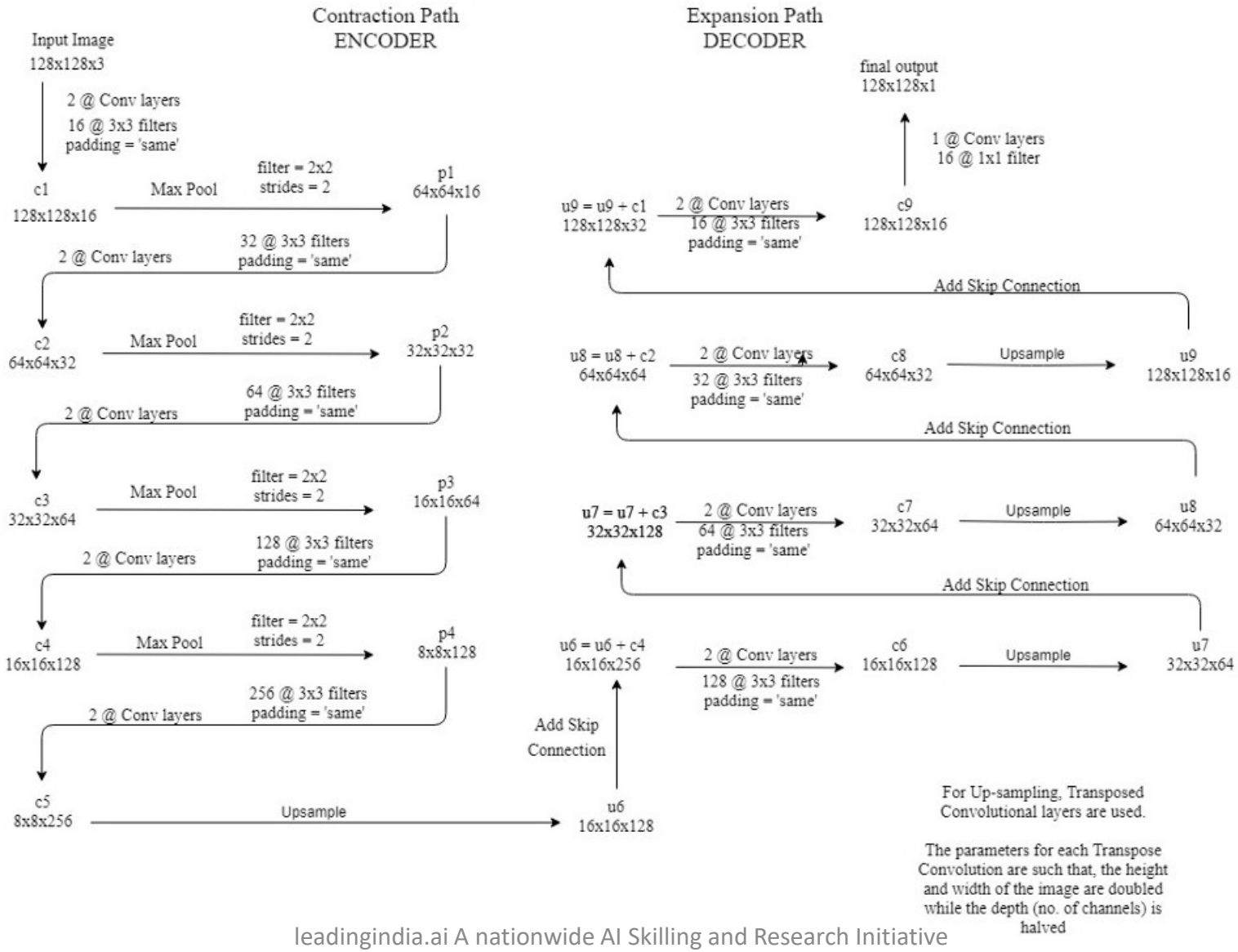
Deconvolution

UNet Architecture

- UNet is a symmetric encoder-decoder network
- Highly used in satellite/medical image segmentation
- Encoder: Contraction path
- Decoder: Expansion path
- Feature detection and location is used at decoder part



UNet Architecture



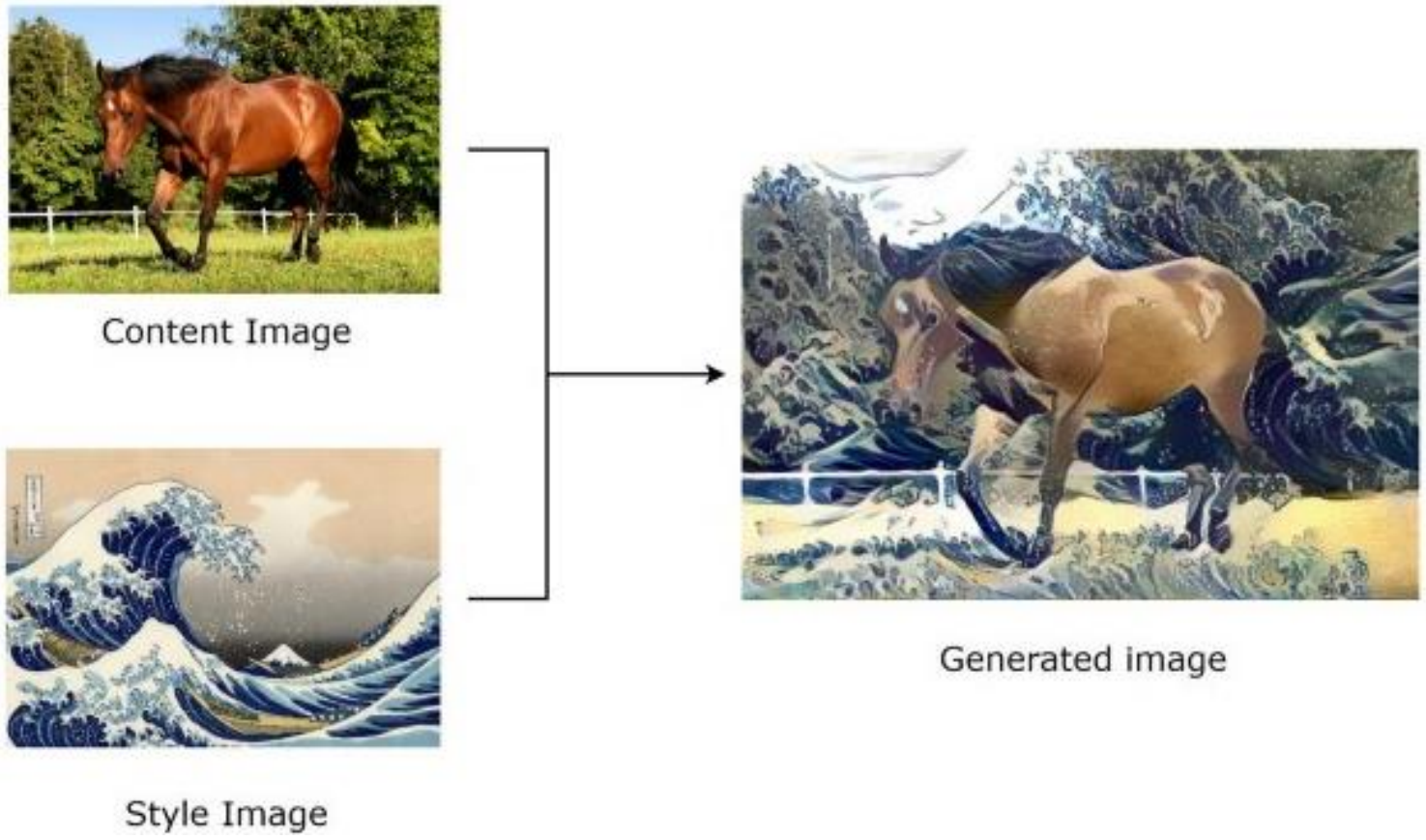
Popular Object Detection /Segmentation Methods

- Focused on Accuracy
  - R-CNN
  - Fast R-CNN
  - Faster R-CNN
  - Mask R-CNN
  - RPN
- Focused on Speed
  - YOLO/YOLOv2/YOLOv3
  - SSD
  - RetinaNet
  - Overfeat
  - M2Det

https://github.com/amusi/awesome-object-detection

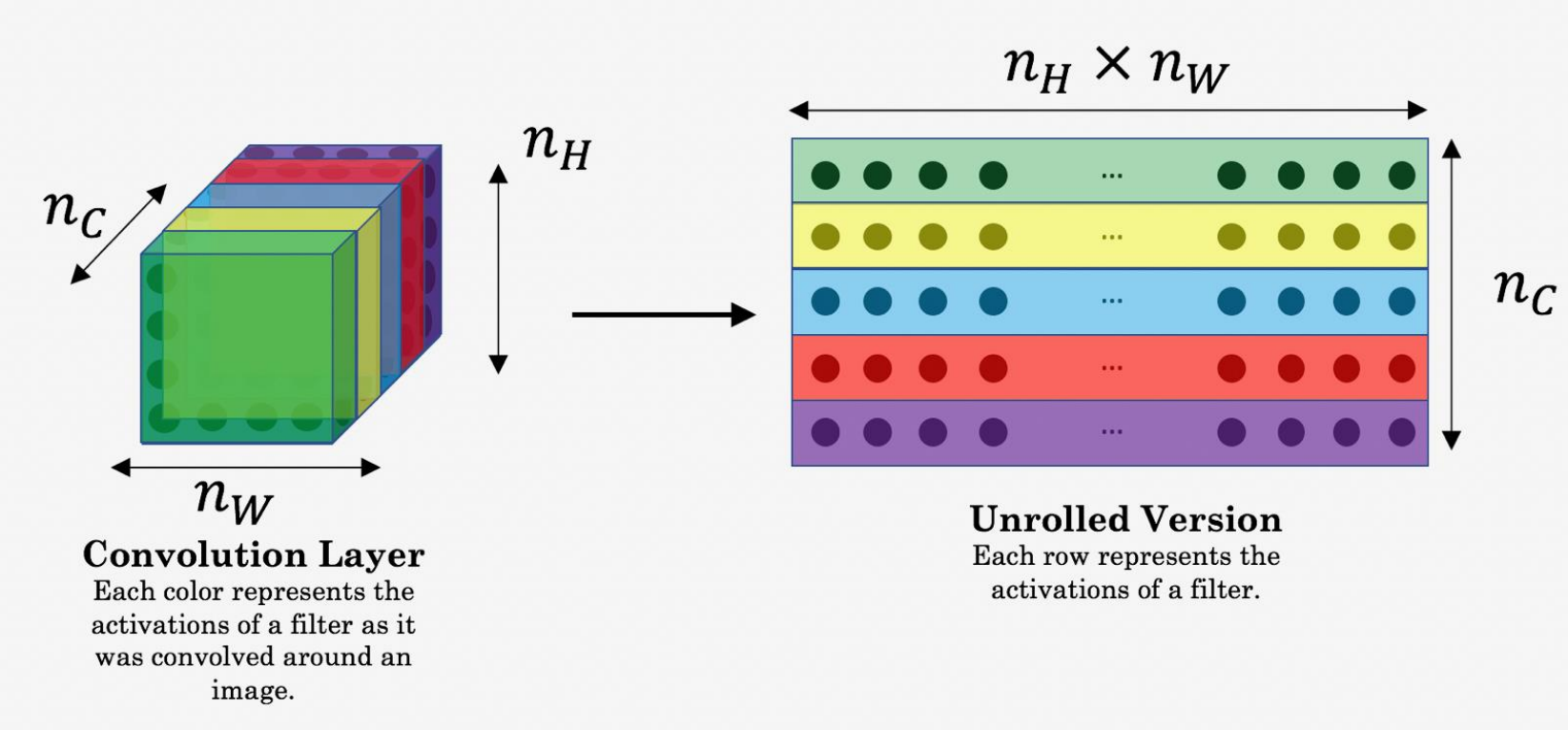
https://github.com/mrgloom/awesome-semantic-segmentation

Neural Style Transfer



- Cost Function $J(G) = \alpha J_{content}(C, G) + \beta J_{style}(S, G)$

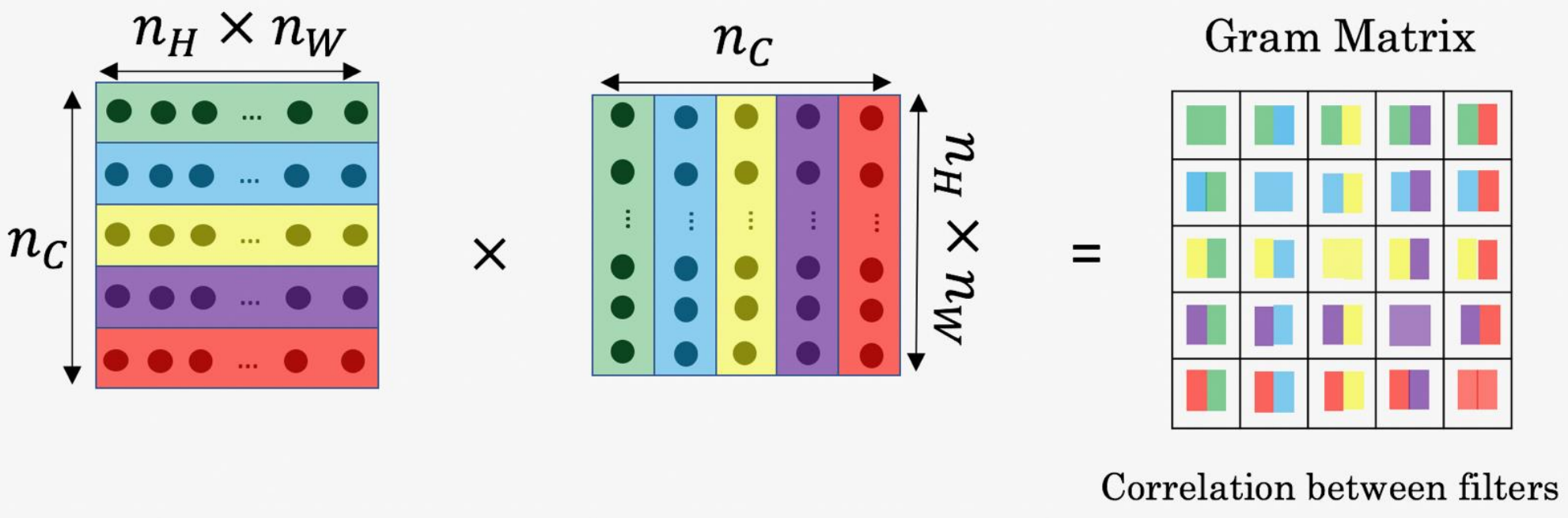
Neural Style Transfer



- Content Cost Function

$$J_{content}(C, G) = \frac{1}{4 \times n_H \times n_W \times n_C} \sum_{\text{all entries}} (a^{(C)} - a^{(G)})^2$$

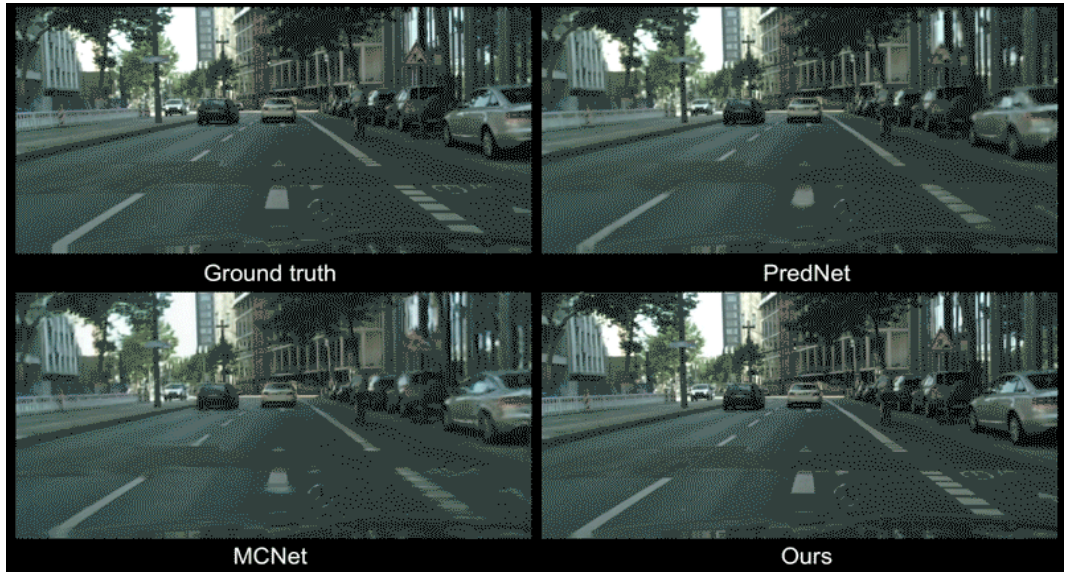
Neural Style Transfer

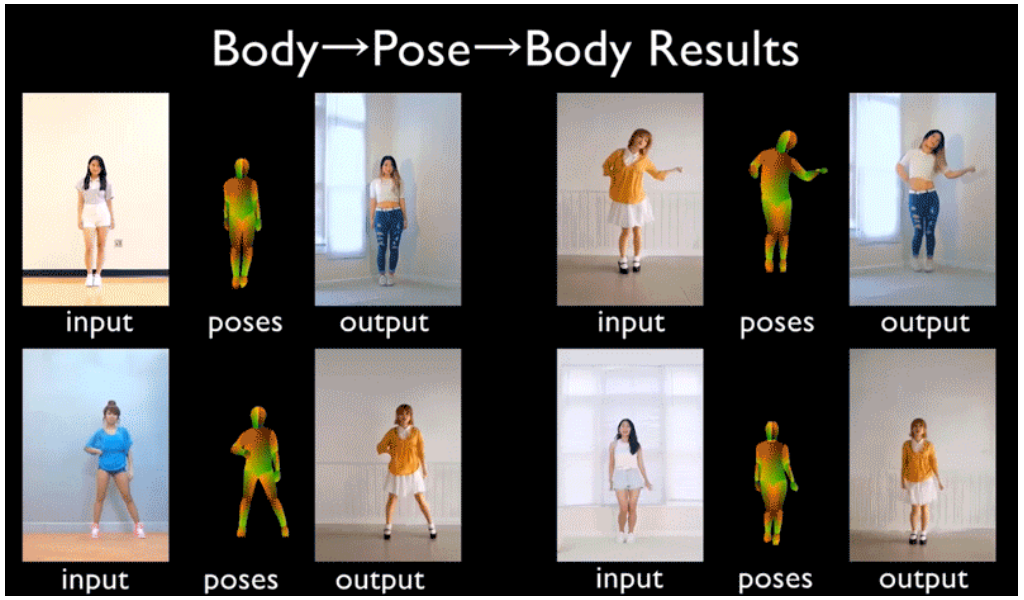


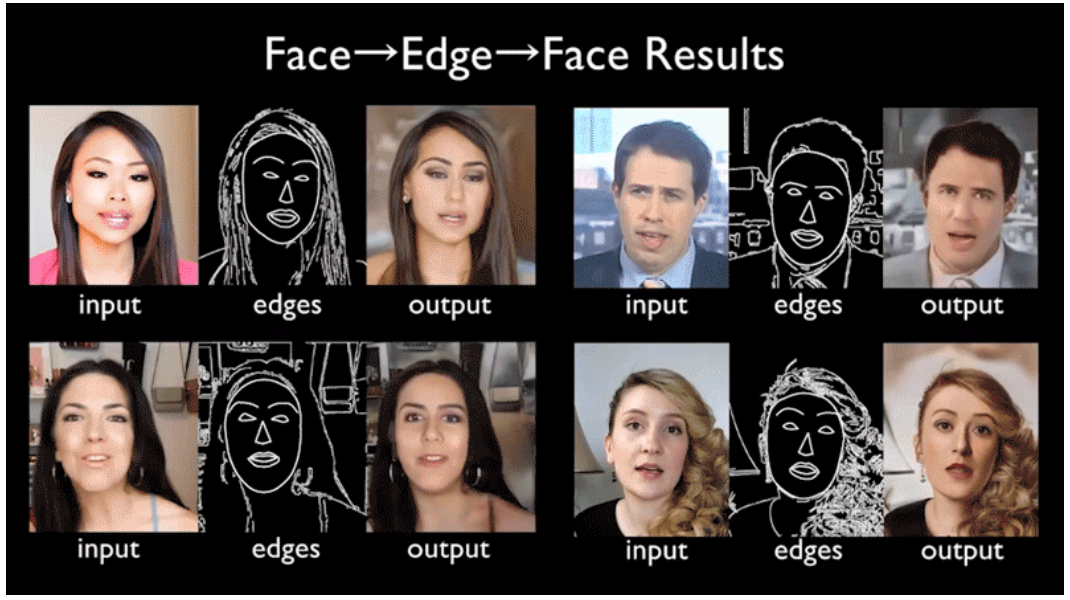
- Style Cost Function

$$J_{style}^{[l]}(S,G) = \frac{1}{4 \times {n_C}^2 \times (n_H \times n_W)^2} \sum_{i=1}^{n_C} \sum_{j=1}^{n_C} (G_{ij}^{(S)} - G_{ij}^{(G)})^2$$

Video to Video Synthesis

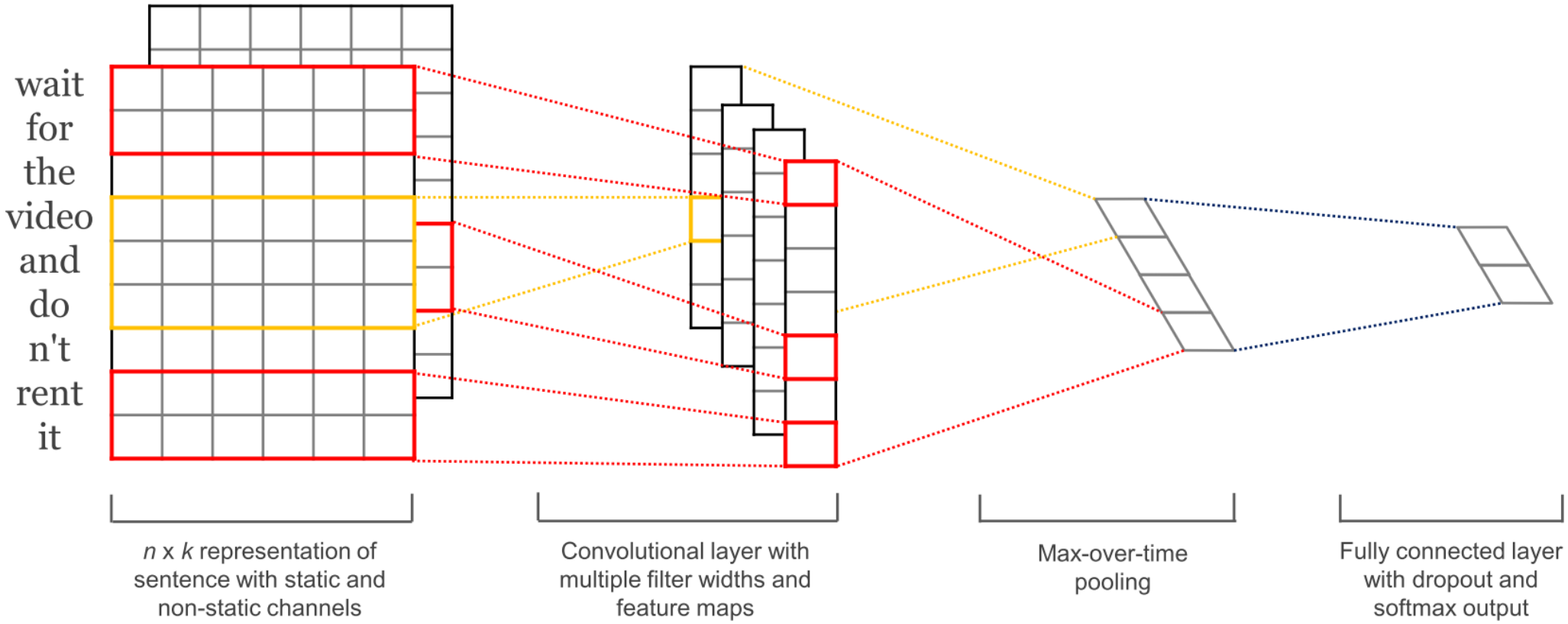






CNN for Text Classification

- 1D Convolutions and 1D pooling can be applied to one-hot vector or word embedding vector for text classification task

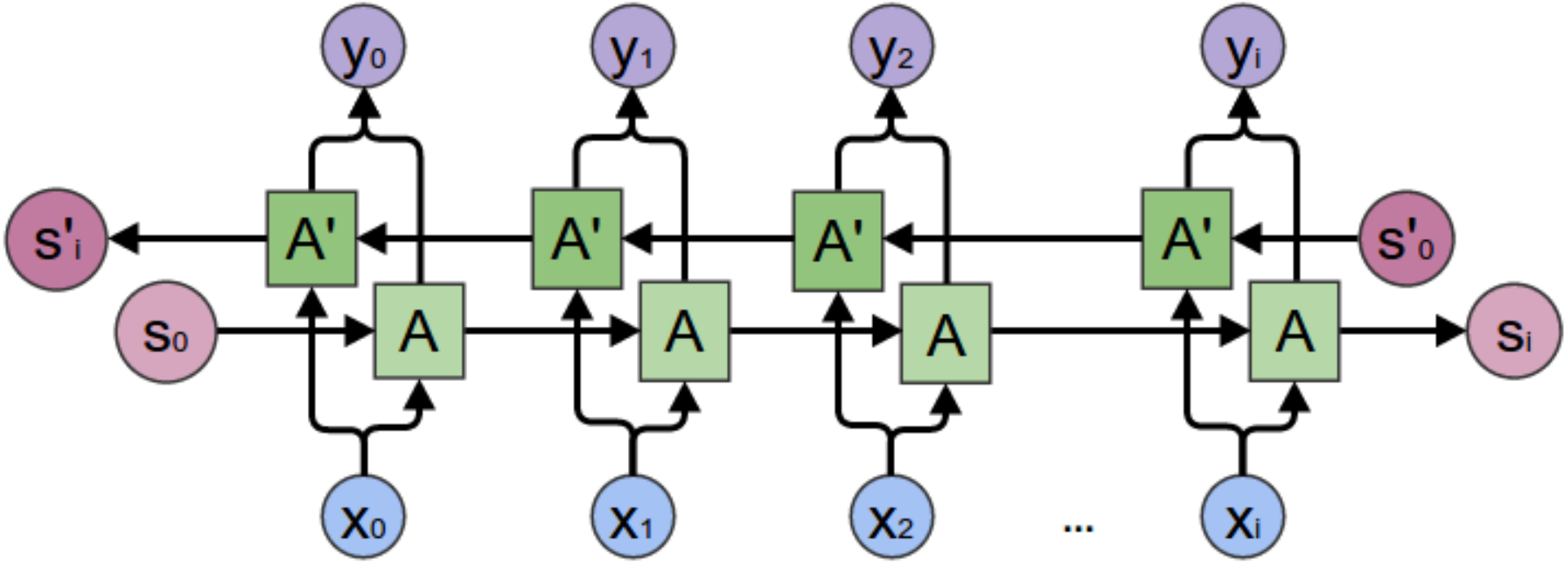


Bidirectional RNNs

“He said, Teddy bears are on sale” and “He said, Teddy Roosevelt was a great President”.

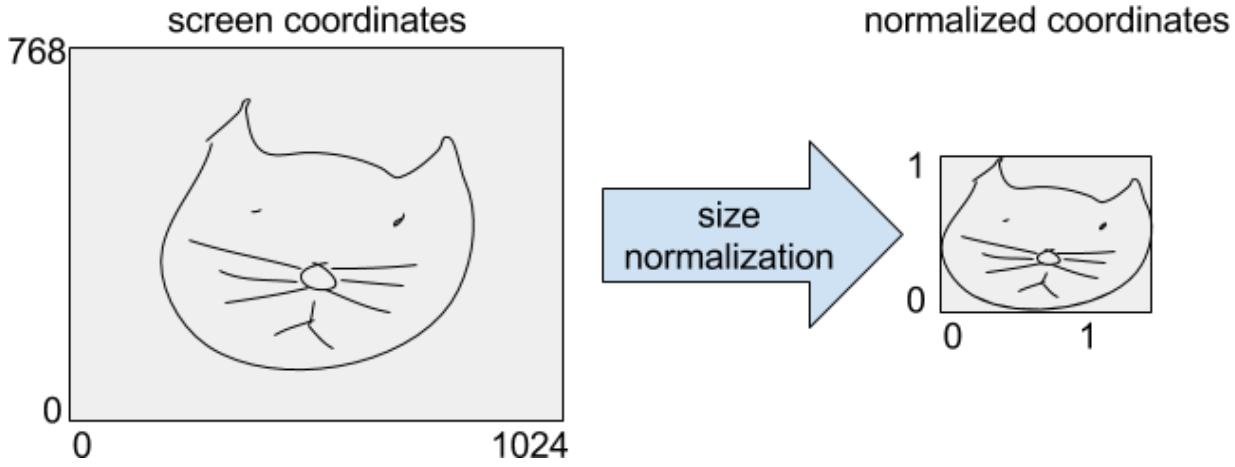
In the above two sentences, when we are looking at the word “Teddy” and the previous two words “He said”, we might not be able to understand if the sentence refers to the President or Teddy bears.

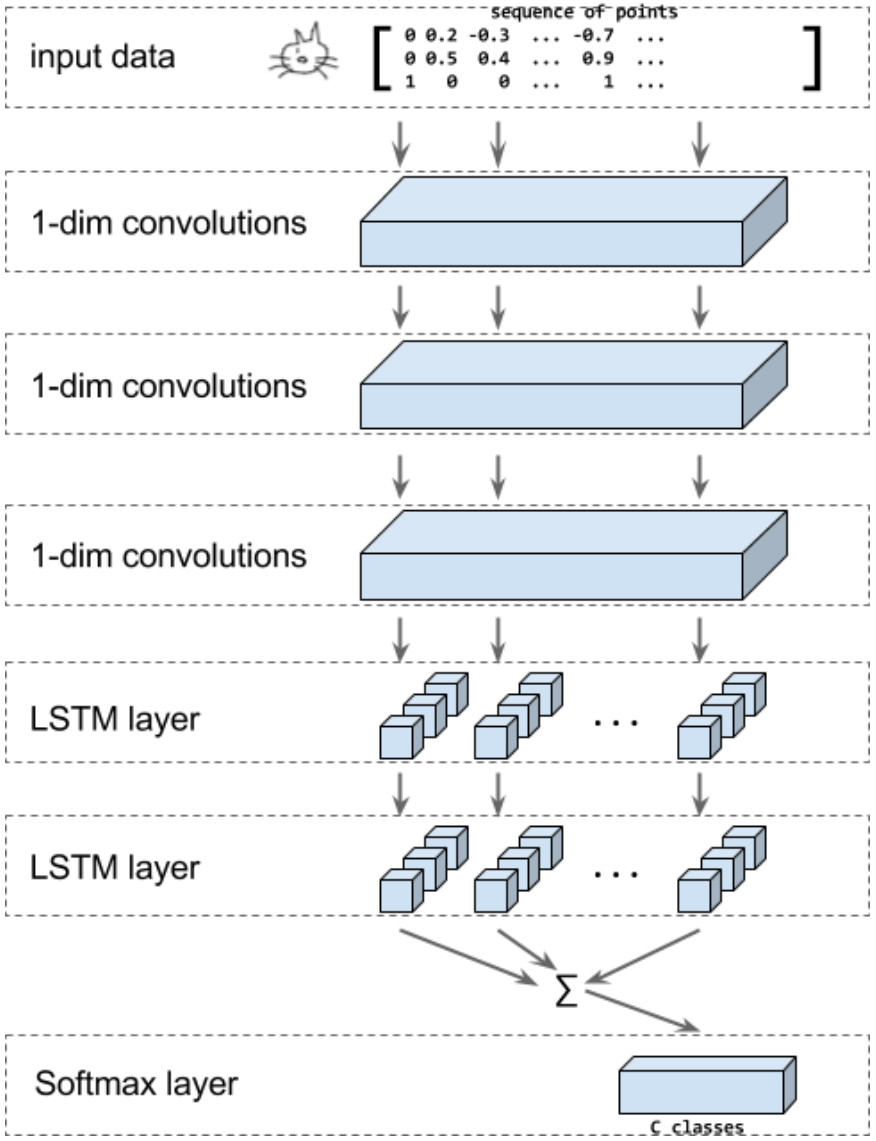
Therefore, to resolve this ambiguity, we need to look ahead. This is what Bidirectional RNNs accomplish.



1D Conv + RNN for Drawing Classification

- 1D convolution can be applied to the sequence of points then applying sequential learning using RNN
- Used in Quick, Draw! of Google

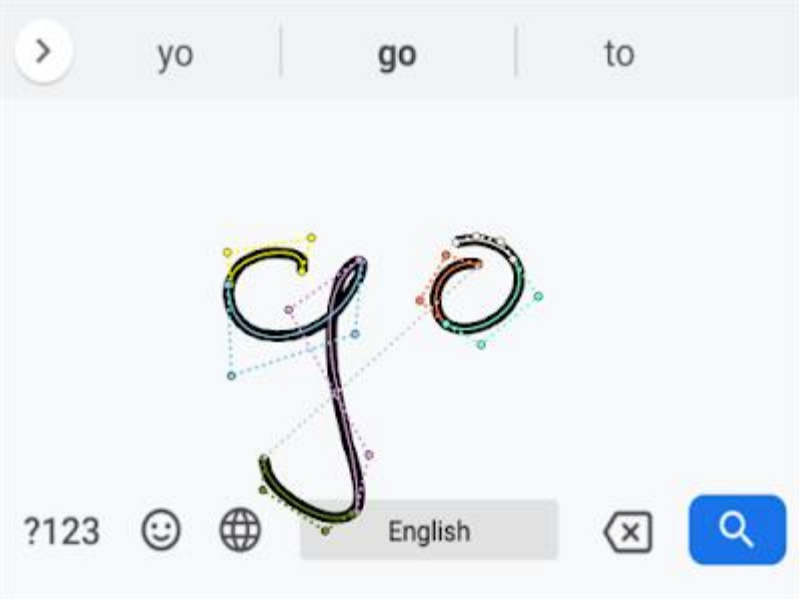


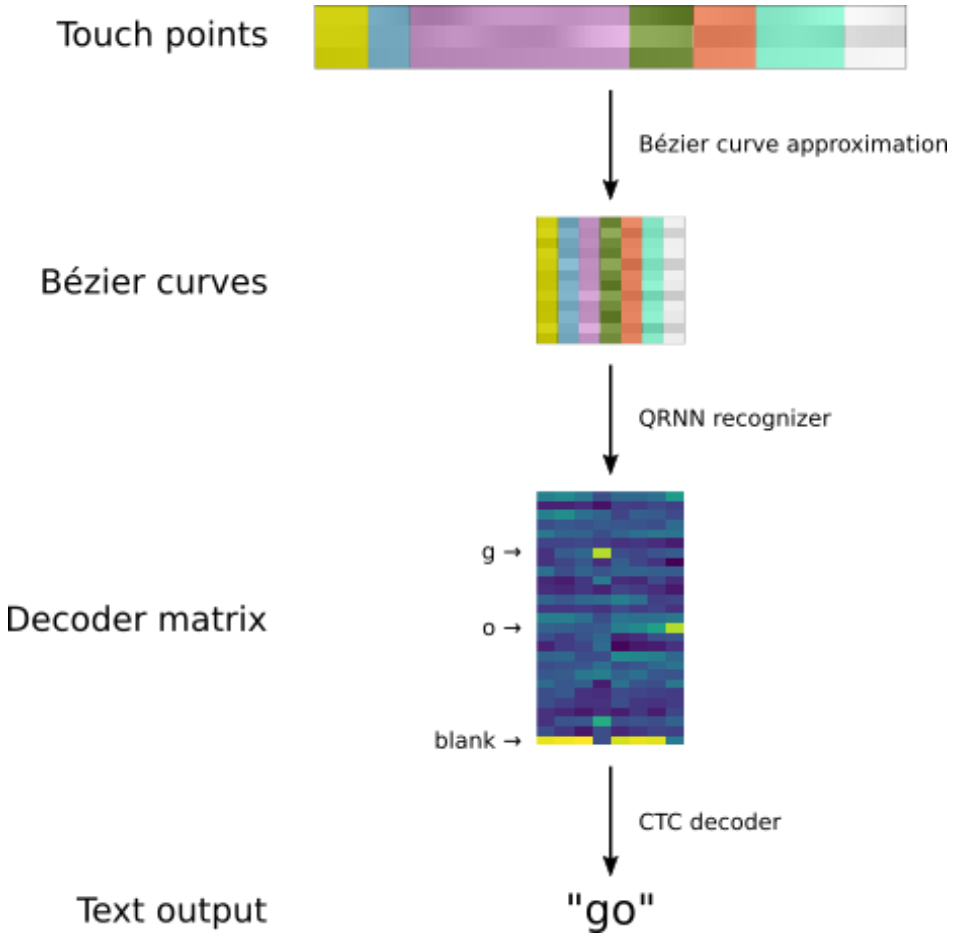


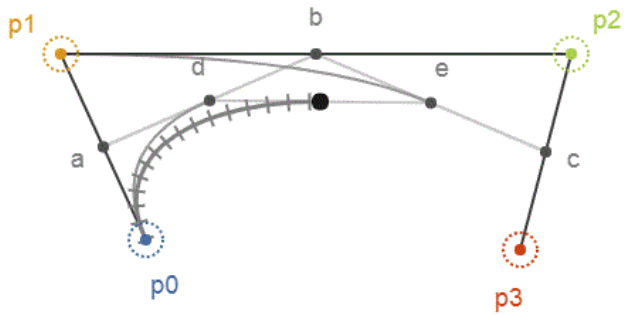
https://quickdraw.withgoogle.com/

RNN based Handwritten Recognition

Google Keyboard
Gboard 2019



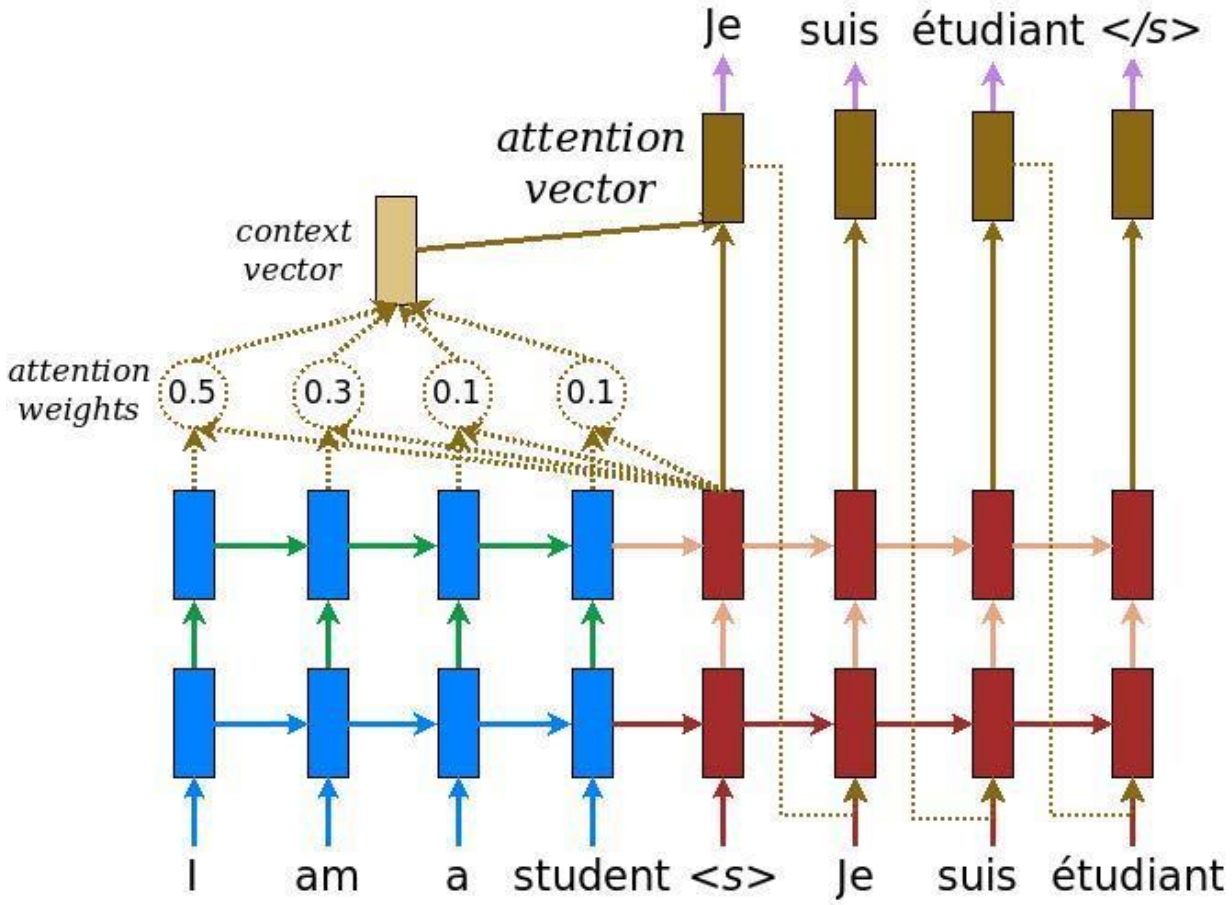


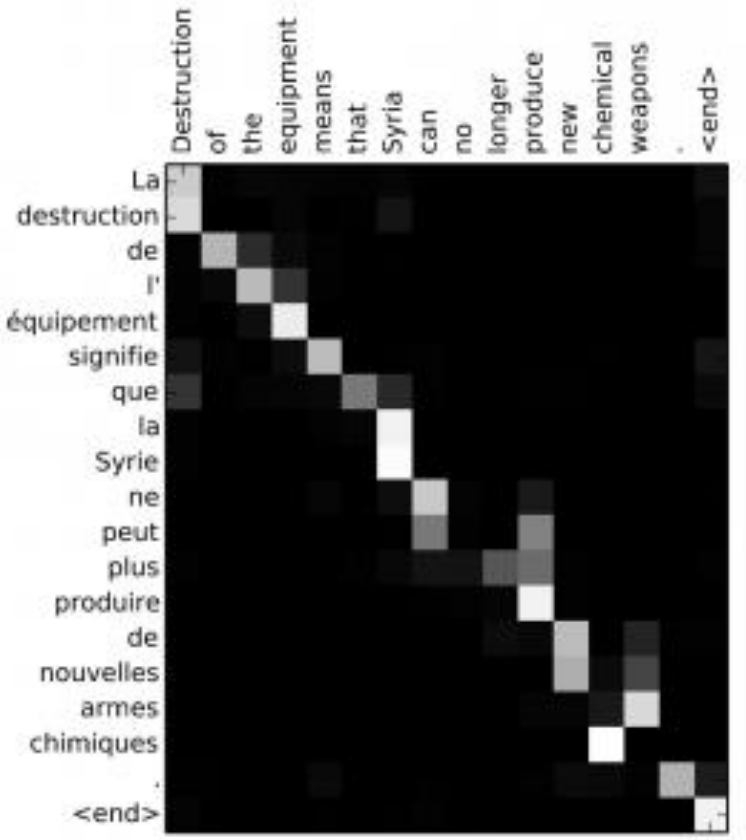


https://ai.googleblog.com/2019/03/rnn-based-handwriting-recognition-in.html

Attention Model

- Attention mechanism helps us to focus on specific parts of input to generate specific output.

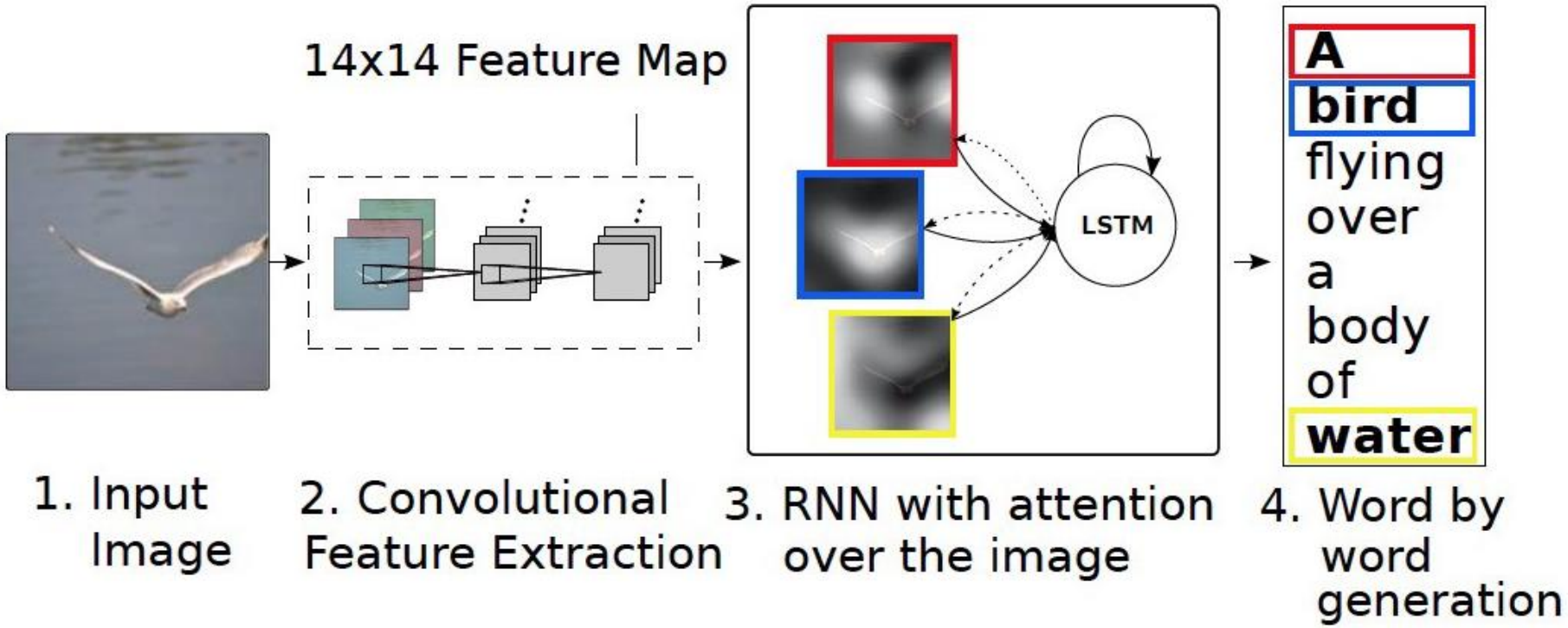




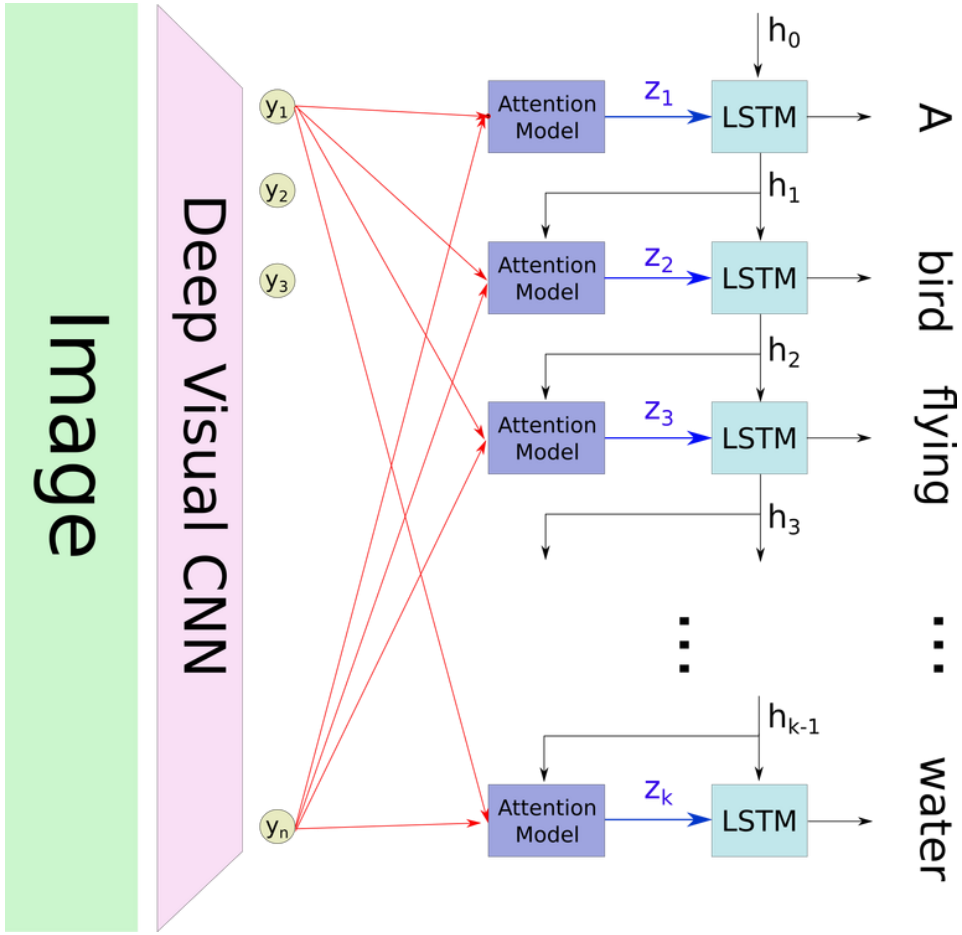
Attention Weight Matrix

Attention Model for Image Captioning

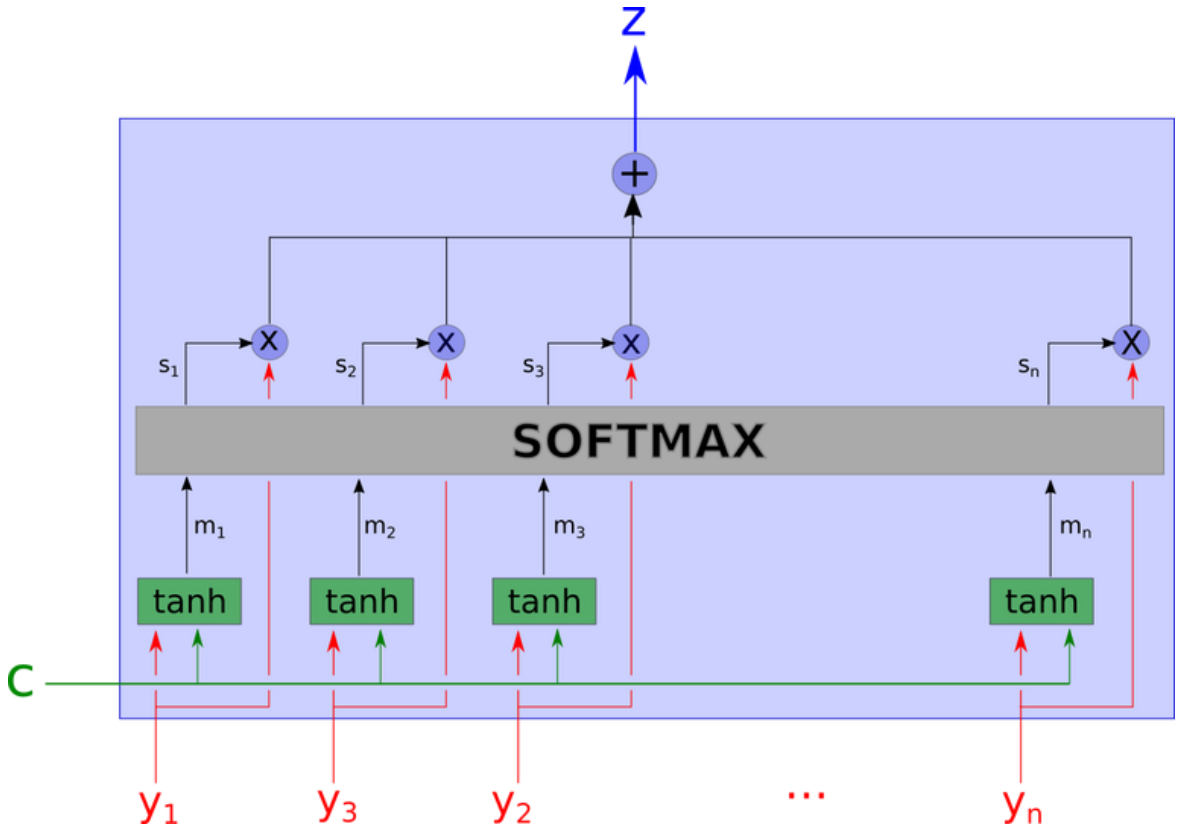
- Attention model in images helps to focus on specific parts of image



Attention Model for Image Captioning

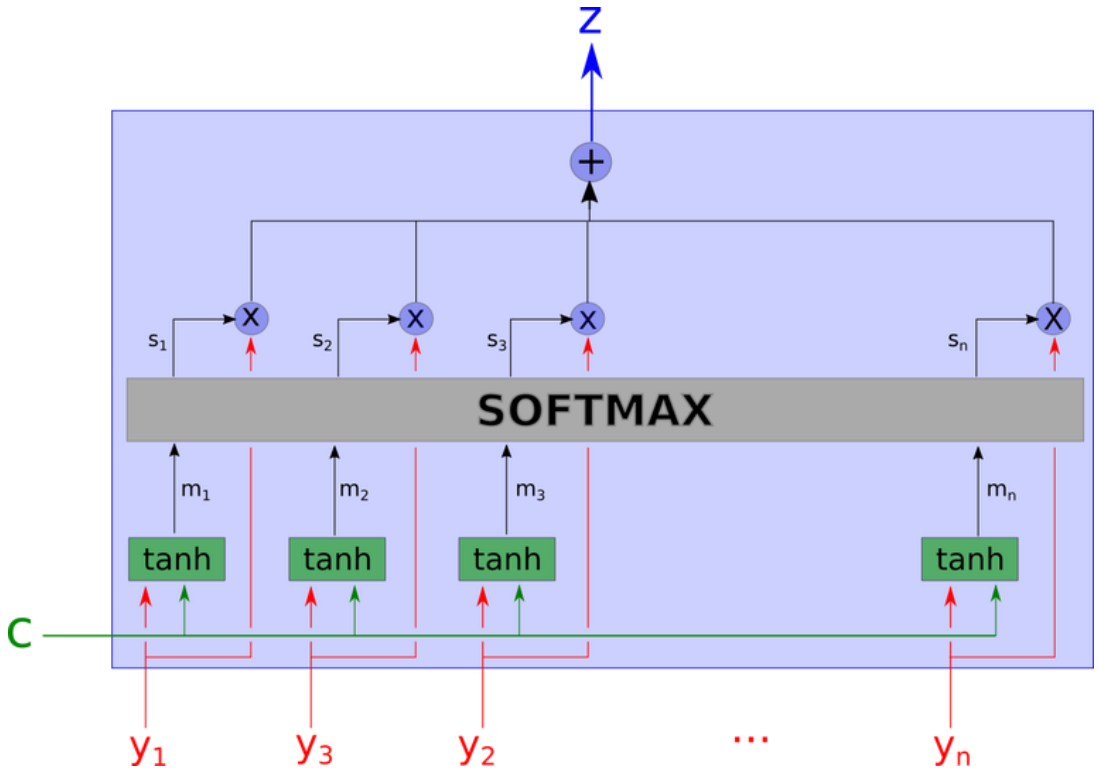


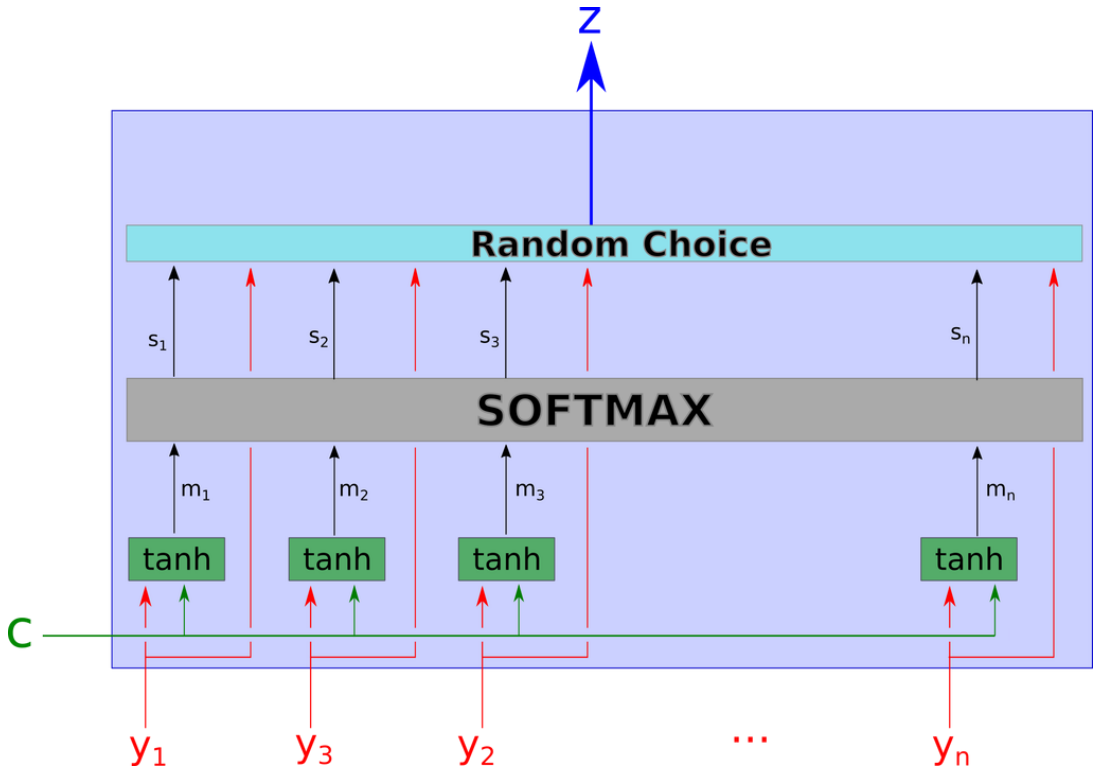
Attention mechanism for Image captioning



Attention Model

Soft and Hard Attentions





Soft Attention

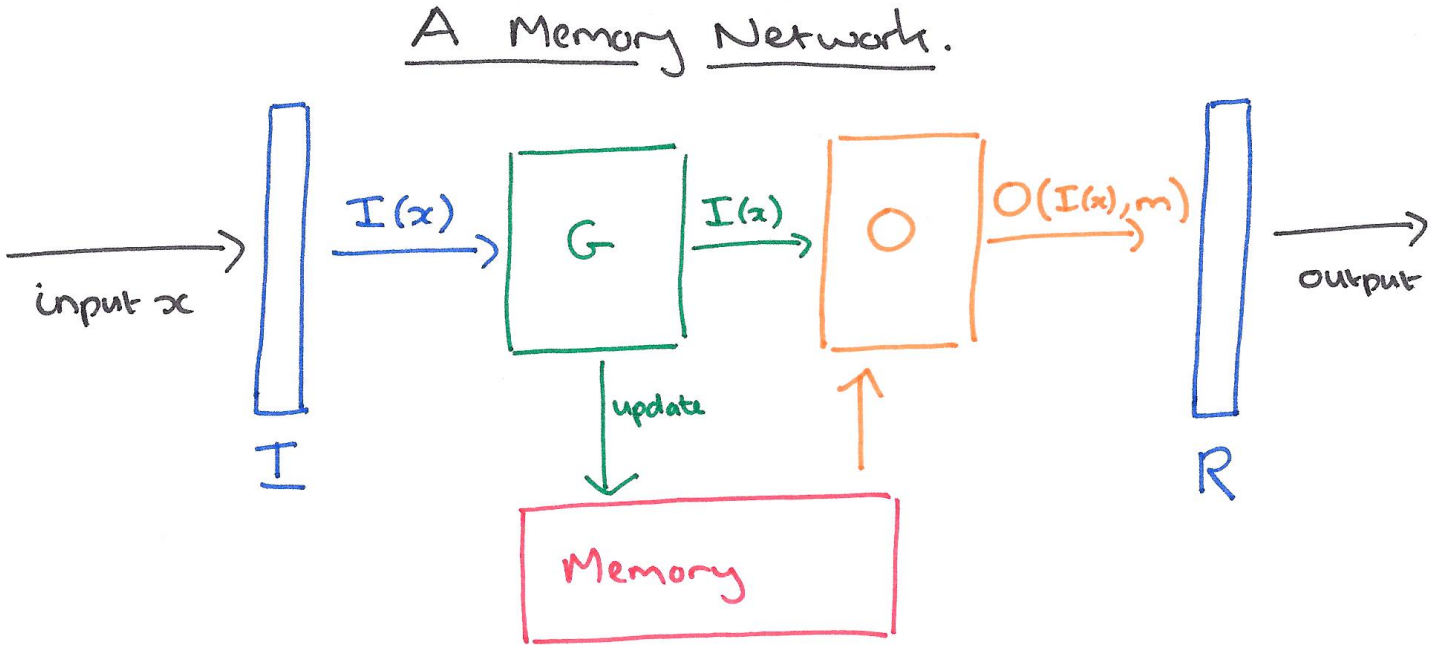
- Deterministic
- Attention as weighted sum of regions

Hard Attention

- Stochastic
- Attention as random region

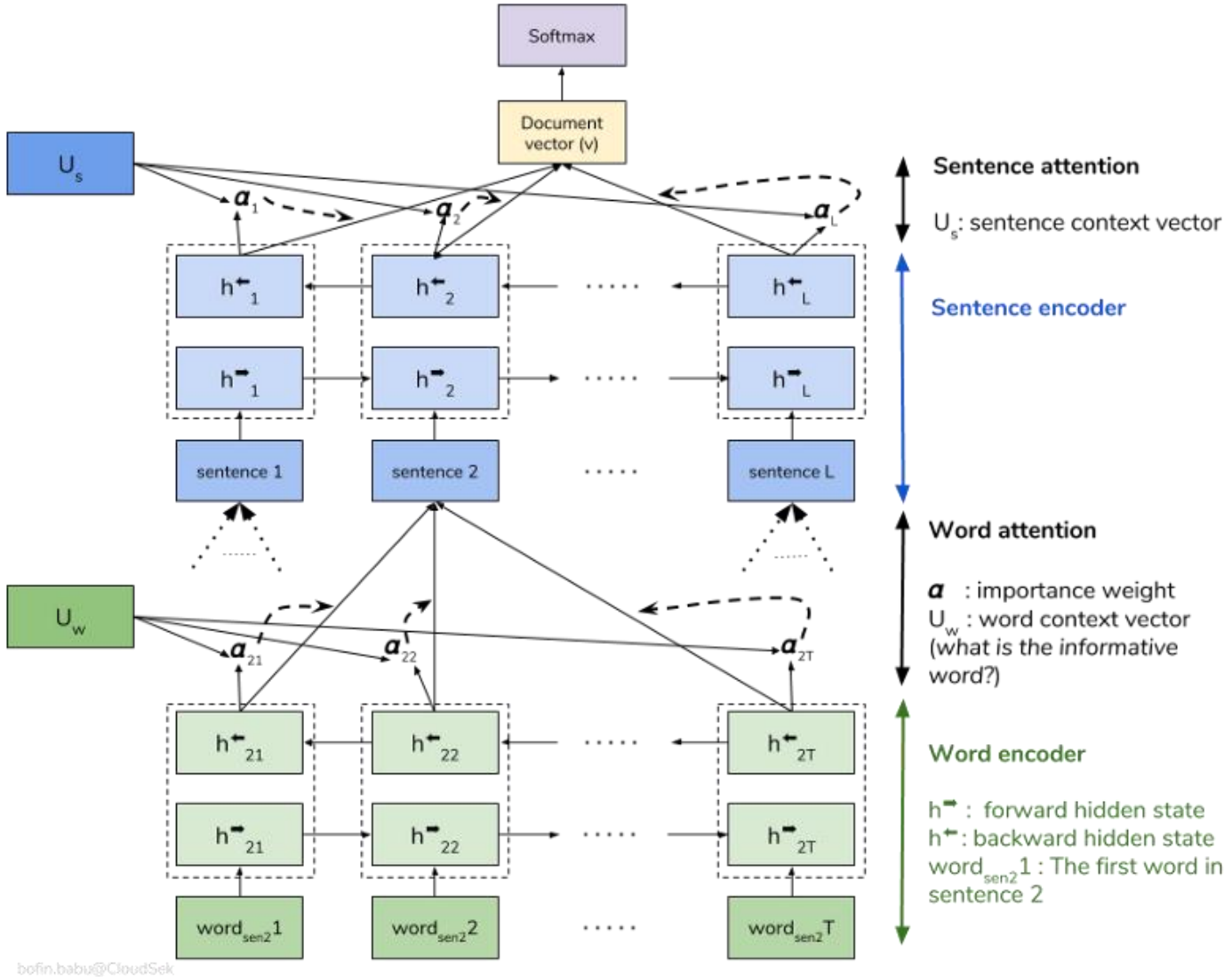
Memory Network

- Memory network work with external data storage, useful for like mapping questions as input to answers stored in that external memory.
- Rather than surfacing the relevant features of an immediate experience, attention can pull a distant episode from the past, as encoded in memory



Text Classification using Hierarchical Attention Network (HAN)

- Idea behind HAN is “Words make sentences and sentences make documents”
- The intent is to derive sentence meaning from the words and then derive the meaning of the document from those sentences.
- Word attention model generates which part of the input words to be focused in a sentence.
- Sentence attention model generates which sentences to be focused to generate document class

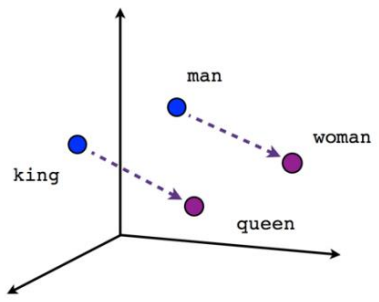


Word Embeddings

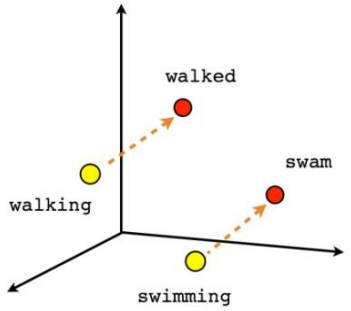
Dimensions

Word vectors				
dog	-0.4	0.37	0.02	-0.34
cat	-0.15	-0.02	-0.23	-0.23
lion	0.19	-0.4	0.35	-0.48
tiger	-0.08	0.31	0.56	0.07
elephant	-0.04	-0.09	0.11	-0.06
cheetah	0.27	-0.28	-0.2	-0.43
monkey	-0.02	-0.67	-0.21	-0.48
rabbit	-0.04	-0.3	-0.18	-0.47
mouse	0.09	-0.46	-0.35	-0.24
rat	0.21	-0.48	-0.56	-0.37

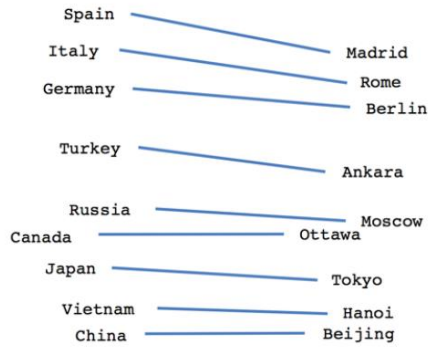
animal
domesticated
pet
fluffy



Male-Female

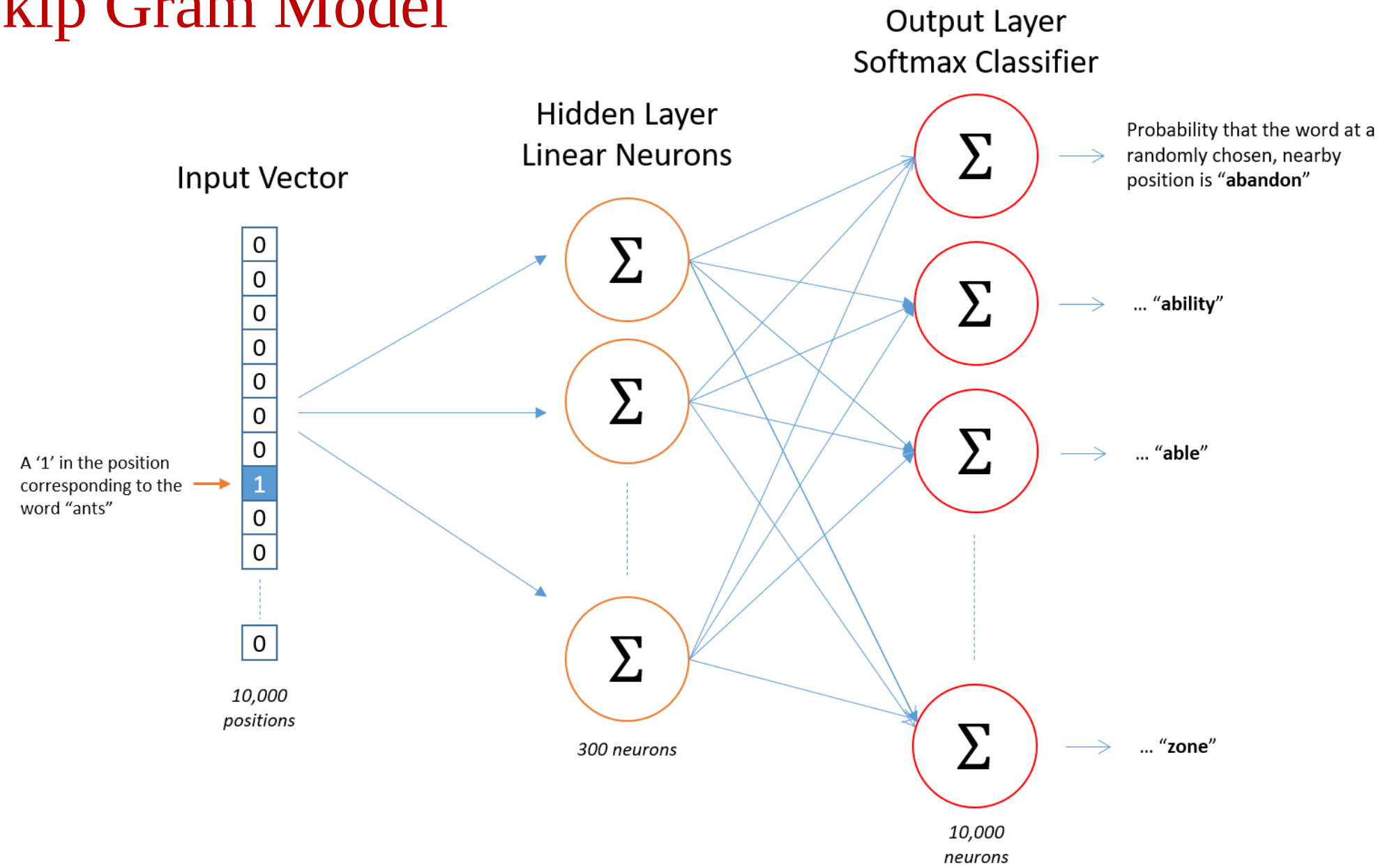


Verb tense

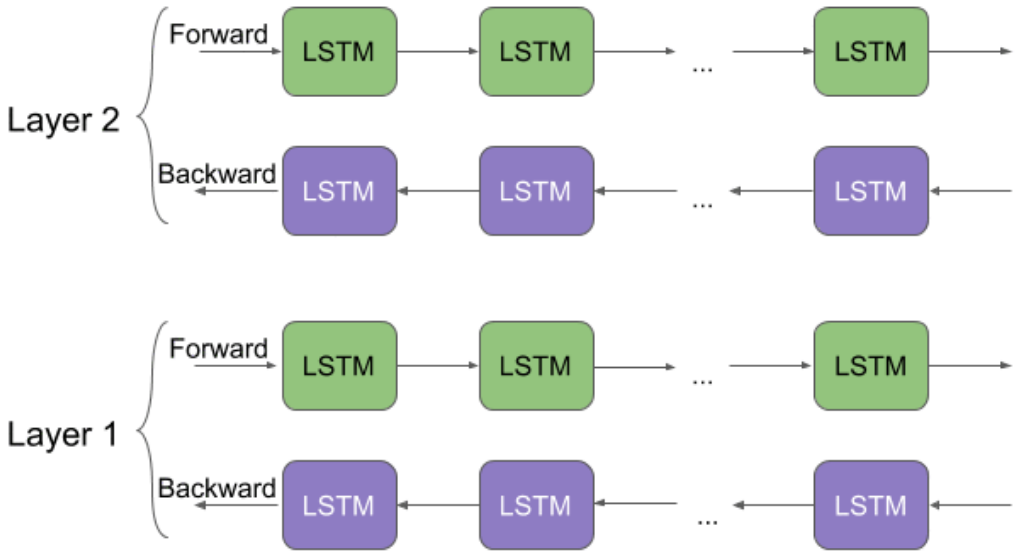


Country-Capital

Skip Gram Model



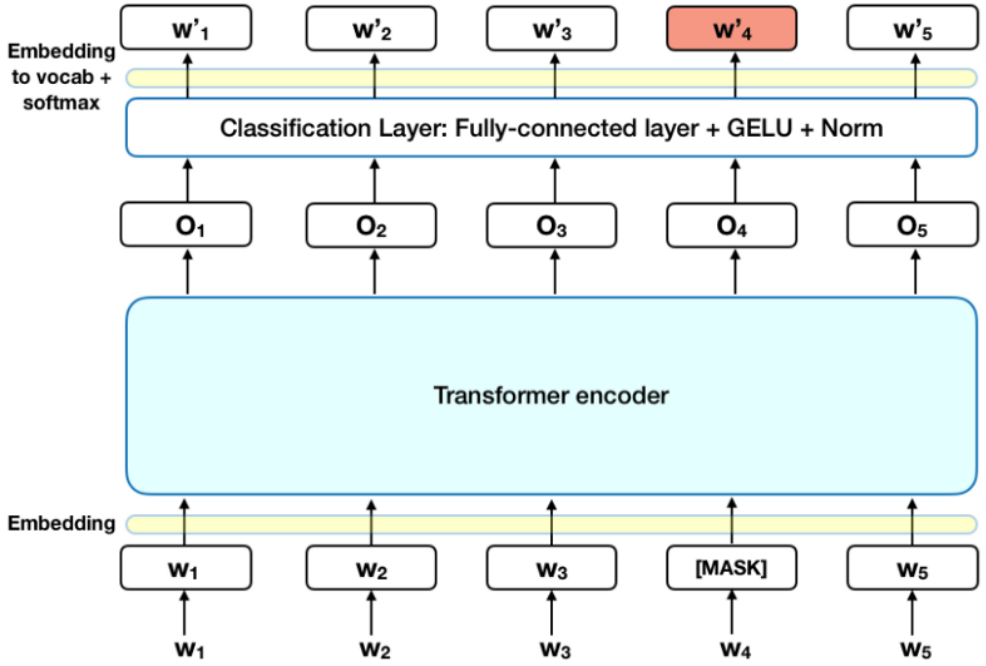
ELMo



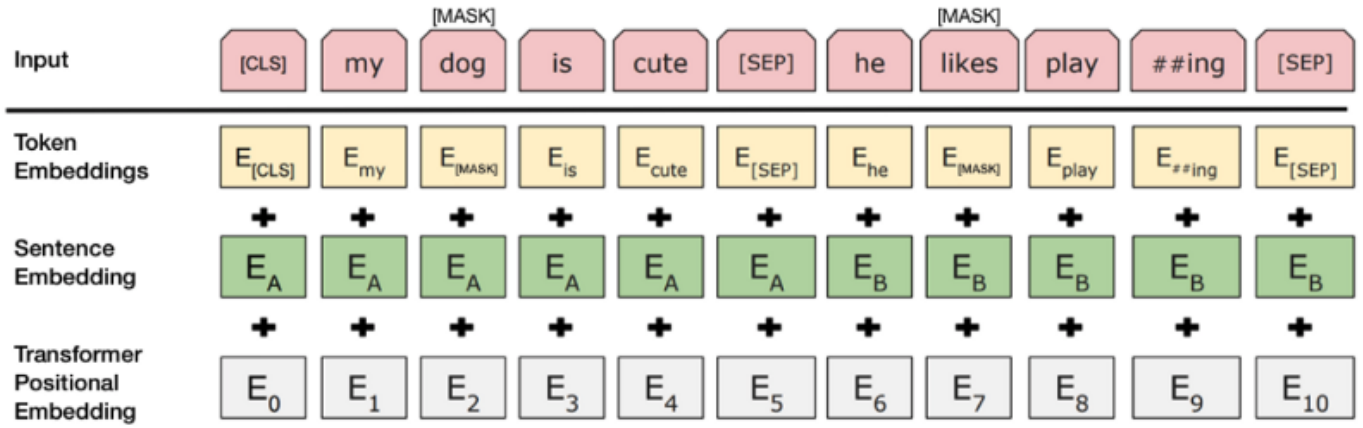
- ELMo (Embeddings from Language Models) is a bidirectional character-level language mode for Word Embedding
- ELMo helps us to handle **Polysemy** wherein a word could have multiple meanings or senses.
  - I ***read*** the book yesterday.
  - Can you ***read*** the letter now?

BERT

- BERT - Bidirectional Encoder Representations from Transformers
- Model is good at generating context based embeddings



Masked LM (MLM)

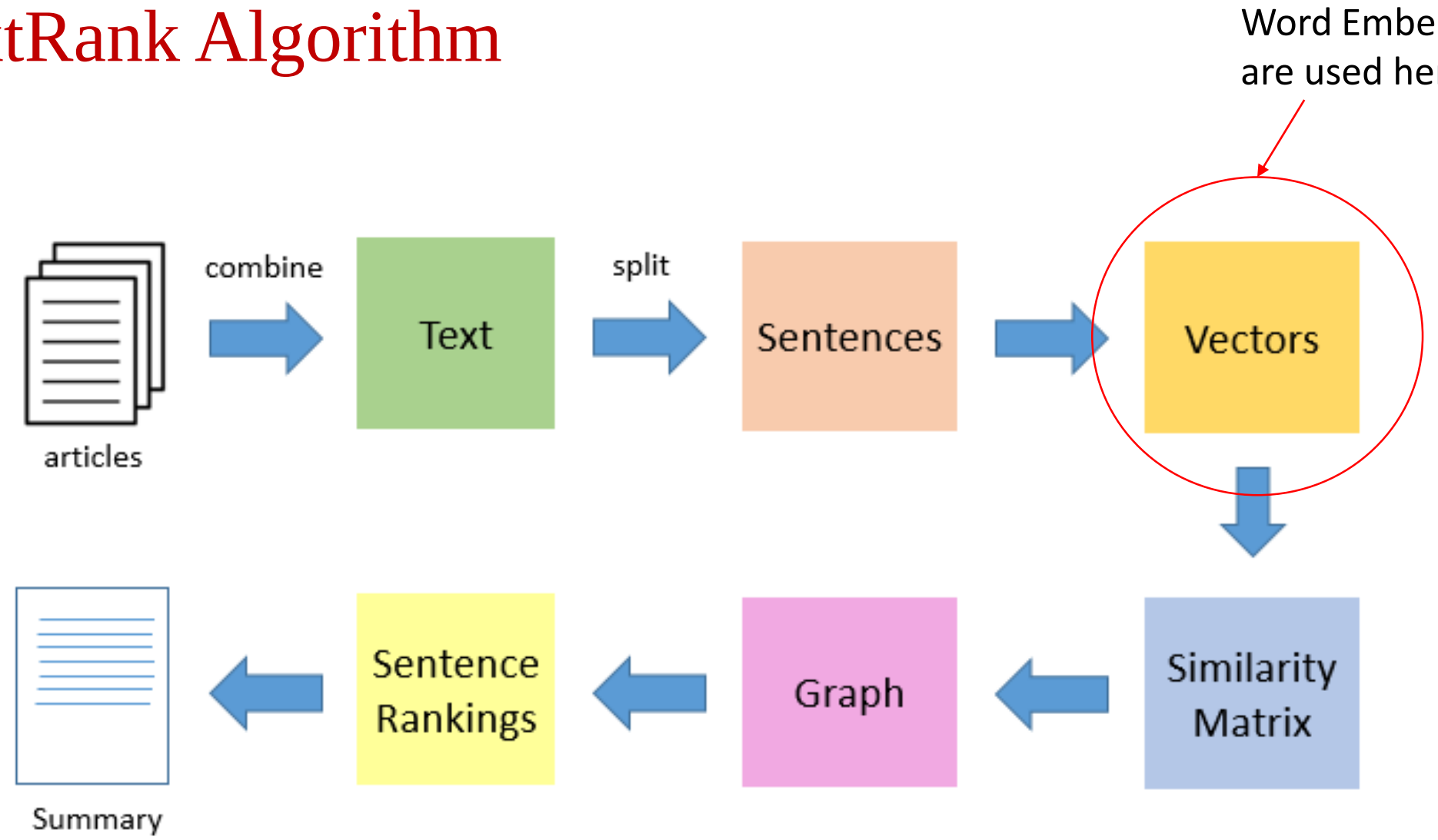


Next Sentence Prediction (NSP)

BERT

- **Masked LM (MLM)**
- Before feeding word sequences into BERT, 15% of the words in each sequence are replaced with a [MASK] token. The model then attempts to predict the original value of the masked words, based on the context provided by the other, non-masked, words in the sequence.
- **Next Sentence Prediction (NSP)**
- In the BERT training process, the model receives pairs of sentences as input and learns to predict if the second sentence in the pair is the subsequent sentence in the original document. During training, 50% of the inputs are a pair in which the second sentence is the subsequent sentence in the original document, while in the other 50% a random sentence from the corpus is chosen as the second sentence. The assumption is that the random sentence will be disconnected from the first sentence.

TextRank Algorithm



MUSE: Multilingual Unsupervised and Supervised Embeddings

- MUSE is a Python library for *multilingual word embeddings*
- Provides state-of-the-art multilingual word embeddings
- Large-scale high-quality bilingual dictionaries for training and evaluation
- https://github.com/facebookresearch/MUSE

Pretrained NLP Models

- **Multi-Purpose NLP Models**
  - ULMFiT
  - Transformer
  - Google's BERT
  - Transformer-XL
  - OpenAI's GPT-2
- **Word Embeddings**
  - ELMo
  - Flair
- **Other Pretrained Models**
  - StanfordNLP

Data Augmentation for NLP

- **Thesaurus:** Replacing words or phrases with their synonyms
- **Word Embeddings**: Using similarity of the word embeddings to find the similar word for replacement. Rather than using static word embeddings, sometimes contextualized word embeddings to replace target word
- **Back Translation:** For example, we want to train a model for translating English→Cantonese, and there is not enough training data for Cantonese. Back-translation is translating Cantonese to English and mixing both original English sentence and back-translated English sentence to train a model
- **Text Generation:** New text data can be generated using rules/models and further added to the dataset

Questions ?

Thanks