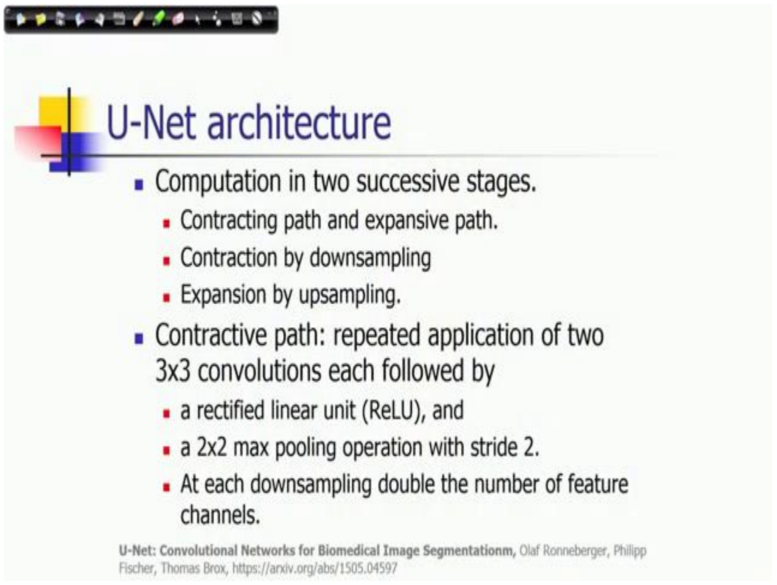
Computer Vision
Prof. Jayanta Mukhopadhyay
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture – 60
Deep Neural Architecture and Applications Part – VI

We are discussing about semantic segmentation using Deep Neural Architecture and in the last lecture I have refer to a very popular segmentation architecture which is called U-Net architecture.

(Refer Slide Time: 00:31)



U-Net architecture

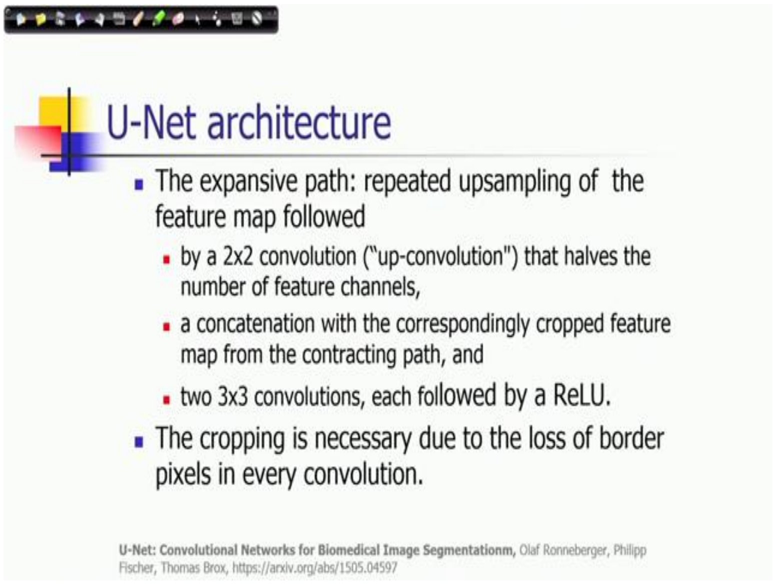
- Computation in two successive stages.
  - Contracting path and expansive path.
  - Contraction by downsampling
  - Expansion by upsampling.
- Contractive path: repeated application of two 3x3 convolutions each followed by
  - a rectified linear unit (ReLU), and
  - a 2x2 max pooling operation with stride 2.
  - At each downsampling double the number of feature channels.

U-Net: Convolutional Networks for Biomedical Image Segmentationm, Olaf Ronneberger, Philipp Fischer, Thomas Brox, https://arxiv.org/abs/1505.04597

So, in this lecture I will give you a brief overview of that architecture. So, in U-Net architecture the competition is carried out in two successive stages. Like the previous architecture of semantic segmentation, there is a path for down sampling and up sampling. Here also we have the path for down sampling and up sampling, we call it as a contracting path and also an expansive path.

So, in the contractive path we have repeated application of two 3×3 convolutions, each of them is followed by a rectified linear unit and they need 2×2 max pooling operation with stride 2. And at each down sampling we also double the number of feature channels. That is how, this architecture is specified you can get the details of this architecture by the paper, what is shown here and you can get it from the internet.

(Refer Slide Time: 01:22)



U-Net architecture

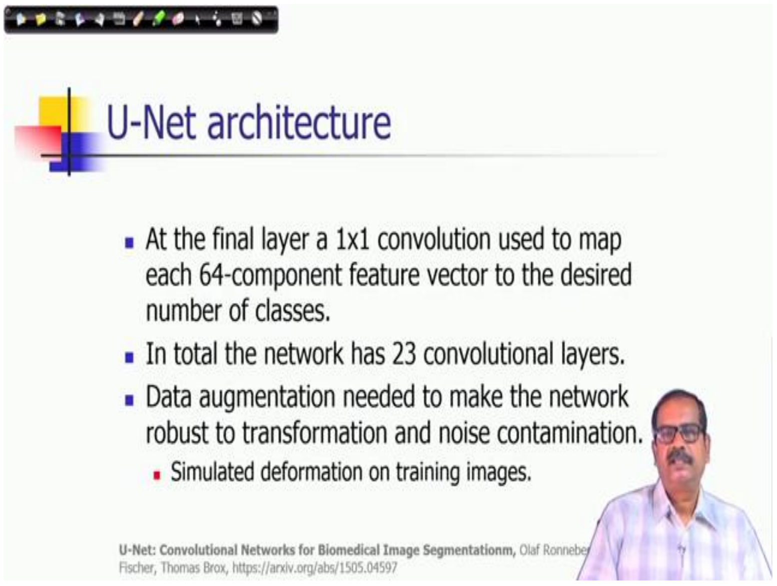
- The expansive path: repeated upsampling of the feature map followed
  - by a 2x2 convolution ("up-convolution") that halves the number of feature channels,
  - a concatenation with the correspondingly cropped feature map from the contracting path, and
  - two 3x3 convolutions, each followed by a ReLU.
- The cropping is necessary due to the loss of border pixels in every convolution.

U-Net: Convolutional Networks for Biomedical Image Segmentationm, Olaf Ronneberger, Philipp Fischer, Thomas Brox, https://arxiv.org/abs/1505.04597

About the expansive path as I was mentioning that it is a repeated up sampling what is used for the feature map, and there is a 2×2 convolution or up convolution is used. And in the contractive path where the number of channels are doubled at every down sampling, in up sampling, that means, in the expansive path rather the feature channels are getting halved. And a concatenation also is there from the contractive paths output, we will show in the diagram.

So, in the expansive path we are using the up sampled features and also the features what is being computed in the contractive path those are concatenated. So, half of them are coming from there and then using them you are basically passing it to the next stage. So, it has three two 3×3 convolutions after doing up sampling and it goes through the stages. Finally, the size of the image or output remains becomes the same as the size of the input, and you require cropping to keep this constraint of size.

(Refer Slide Time: 02:32)


U-Net architecture

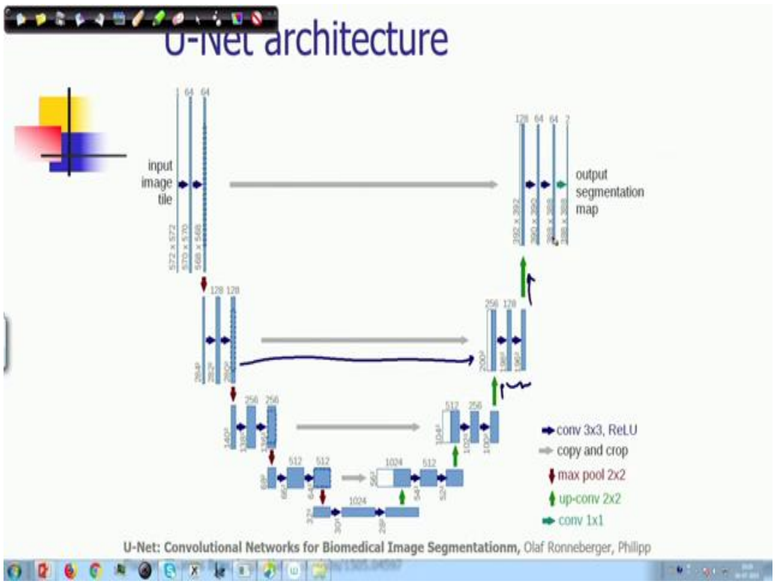
- At the final layer a 1x1 convolution used to map each 64-component feature vector to the desired number of classes.
- In total the network has 23 convolutional layers.
- Data augmentation needed to make the network robust to transformation and noise contamination.
  - Simulated deformation on training images.

U-Net: Convolutional Networks for Biomedical Image Segmentationm, Olaf Ronnebe
Fischer, Thomas Brox, https://arxiv.org/abs/1505.04597

And at the final layer when you have generated all these pixels at the output, then for each pixel you classify those pixels into a number of classes. So, every pixel at the final layer, it is represented by a feature of 64 feature; 64 dimensional feature vector. And those 64 dimensional feature vectors becomes a input to every classifier.

So, in total there are 23 convolutional layers and in particular this work, they have it is used applied for medical image segmentation or segmentation of microscopic images and there are data augmentations which are used to increase the data pool, for example, they have used simulated deformations on training images.

(Refer Slide Time: 03:23)



U-Net architecture

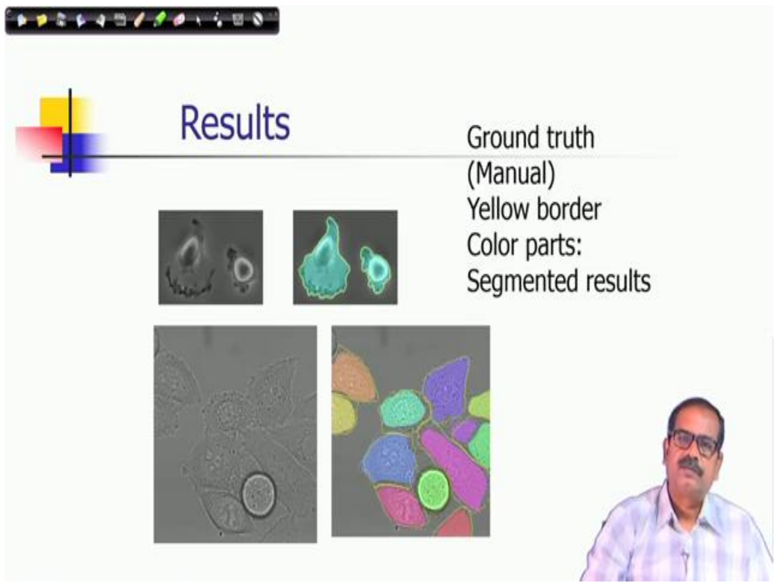
U-Net: Convolutional Networks for Biomedical Image Segmentationm, Olaf Ronneberger, Philipp

So, this is a schematic diagram which is taken from that paper itself. It is a U-Net architecture and you can see that how the contractive path it is down sampling images of different sizes and the number of channels are getting doubled there. And then in the expensive path, you have up sampling images.

And not only that as I was mentioning there is a concatenation of features. So, these features it is concatenated at this stage. So, half of this is coming from this stage and half of them is coming here and it become once again, this becomes a input to the next two convolution layers. Then again it is up sampled and it is performed in the same fashion.

So, this is the operation, what is being done in the U-Net architecture and you using the input or specification you train it, and then once the training is over then you can use it for the purpose of semantic segmentation.

(Refer Slide Time: 04:23)



Results

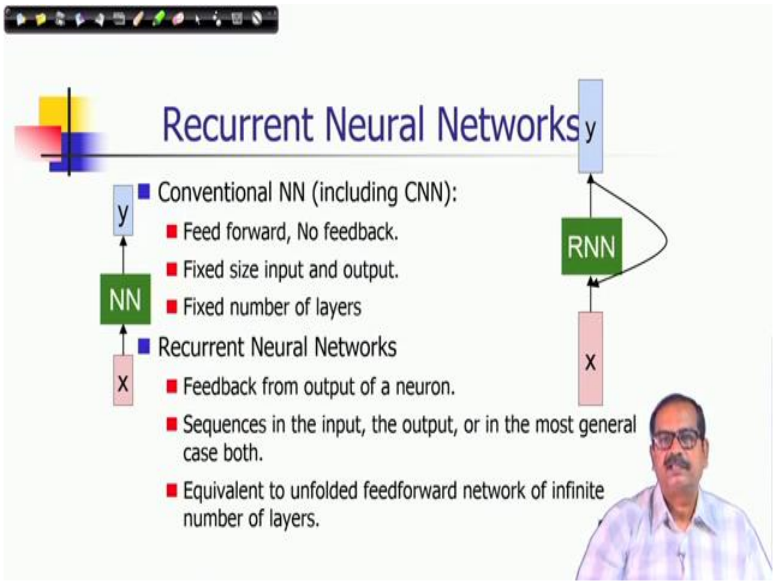
Ground truth (Manual) Yellow border

Color parts: Segmented results

So, some of the results which are there in that paper I am just producing them. So, you can see that these are microscopic images. The top one is showing the segmentation of objects that these are the cells and they are the yellow border is the manually segmented that is the ground truth and the colour part is a result. So, it is very close we can see.

The bottom rows and the pairs of images are shown, original image and also segmented image. It is quite interesting and which is shown that it is very nicely it captures the corresponding segments and which collaborates with the ground truth also very close to it.

(Refer Slide Time: 05:11)



Recurrent Neural Networks

- Conventional NN (including CNN):
  - Feed forward, No feedback.
  - Fixed size input and output.
  - Fixed number of layers
- Recurrent Neural Networks
  - Feedback from output of a neuron.
  - Sequences in the input, the output, or in the most general case both.
  - Equivalent to unfolded feedforward network of infinite number of layers.

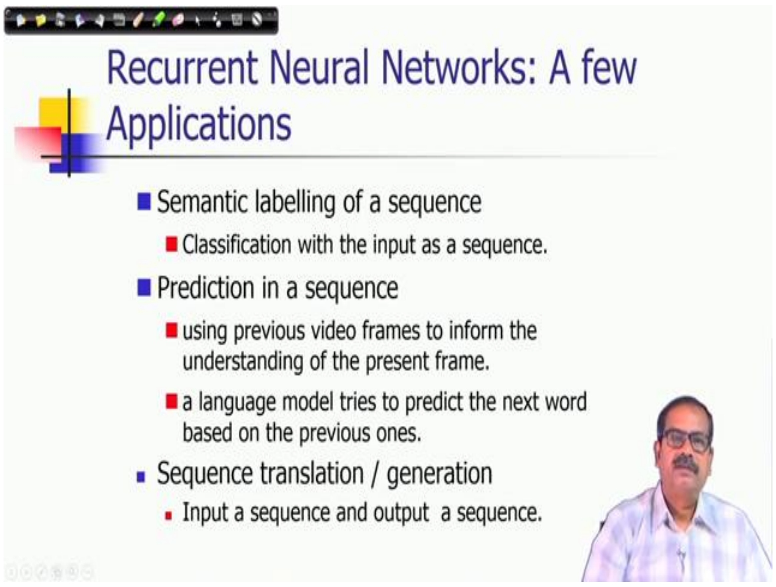
So, next another kind of architecture we will be discussing and we will be considering another types of problems which could be solved using neural architecture, these networks are called Recurrent Neural Networks.

So, let us first understand the difference between recurrent neural networks. For example, the conventional neural network, including CNN there is no feedback, it has been shown in the left side that NN block the green NN block, where the input is x and output is y and there is no connections from the output to the towards the input. Whereas, in recurrent neural networks you have a feedback, you have a connections from the output part to the input block.

So, in the conventional neural network it is a feed forward and there is no feedback. Whereas, in the recurrent neural networks you have feedback from output of a neuron. A conventional neural network it handles fixed size input and output, in the recurrent neural network input could be a variable size. It is a sequences in the input the output or in the most general case both.

Conventional neural network it is a fixed number of layers and since, there is a feedback actually it can simulate an infinite number of feed forward network, it can unfold into an infinite number of layers. So, this is the characteristics of recurrent neural networks.

(Refer Slide Time: 06:47)

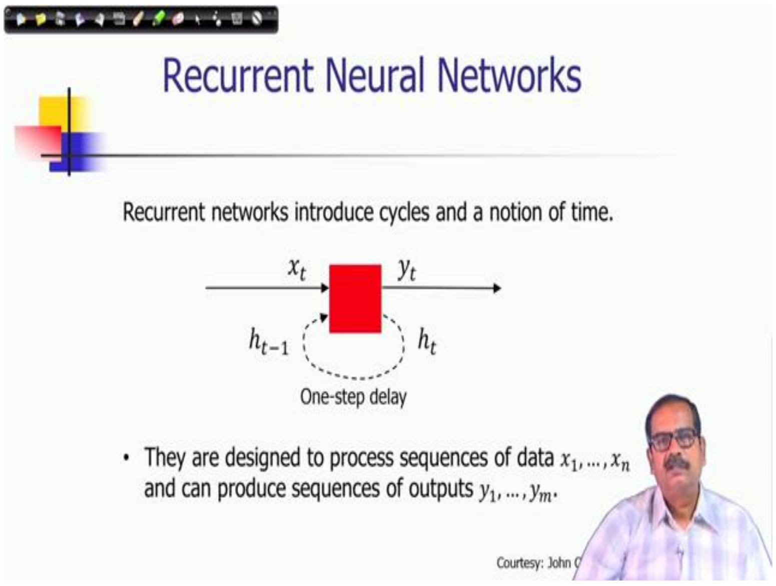


Recurrent Neural Networks: A few Applications

- Semantic labelling of a sequence
  - Classification with the input as a sequence.
- Prediction in a sequence
  - using previous video frames to inform the understanding of the present frame.
  - a language model tries to predict the next word based on the previous ones.
- Sequence translation / generation
  - Input a sequence and output a sequence.

Some of the applications of recurrent neural networks, let us quickly visit some of those applications, like semantic labeling of a sequence. So, it can classify the input as a sequence, then prediction in a sequence. So, using previous a video frames to inform the understanding of the present frame one kind of example or you can consider the language model, it tries to predict the next word, based on the previous ones. And sequence translation and generation where input is a sequence and output is also a sequence.

(Refer Slide Time: 07:25)



Recurrent Neural Networks

Recurrent networks introduce cycles and a notion of time.

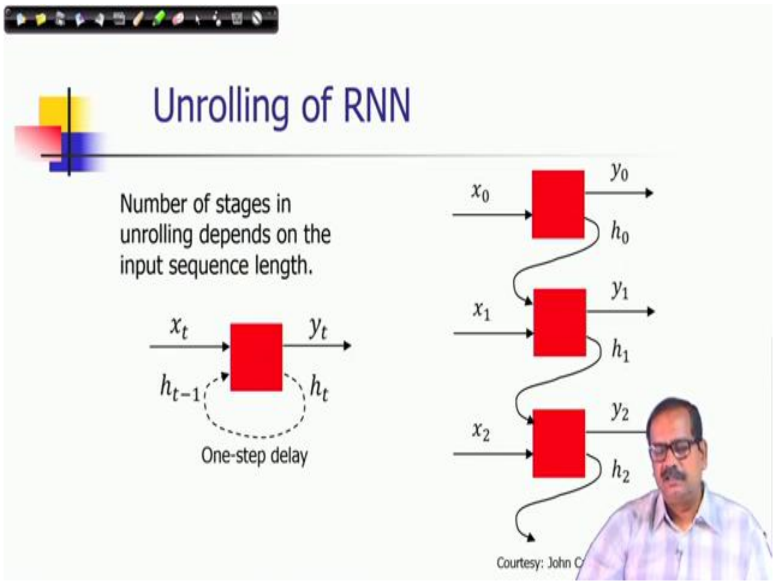
- They are designed to process sequences of data $x_1, \dots, x_n$ and can produce sequences of outputs $y_1, \dots, y_m$.

Courtesy: John C

So, the basic feature of the recurrent neural network is that it introduced cycles and notion of time. So, consider this particular diagram and here the red square is representing a recurrent neural network node. So, input at time t, x_t and then there is output y_t, but the intermediate output from the network which has been also given as a feedback. So, you can see that input is not only x_t even the previous time instances output is also input to this block h_{t-1}.

So, as if there is a one step delay and this is how it works. So, this is used to design two process sequences of data and it can produce sequences of outputs. Because of this feature, that you are introducing a notion of sequence or time and it is kind of a delay in the processing. So, we can consider when you are giving input, input can come as a sequence and when you are producing output, output we will also a sequence.

(Refer Slide Time: 08:47)



Unrolling of RNN

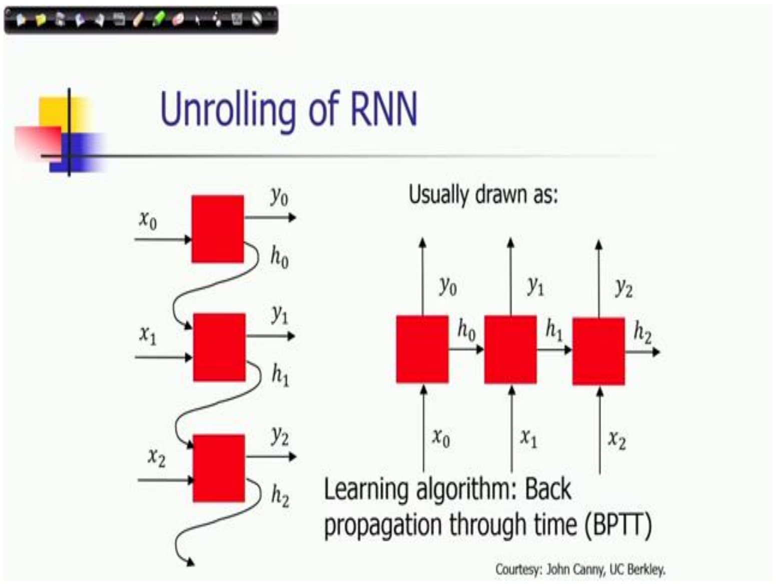
Number of stages in unrolling depends on the input sequence length.

So, to analyze a recurrent neural network, we have to unroll or unfold the neural network. And number of stages of unrolling it depends upon the length of the input sequence.

So, you consider this that suppose you have three; length of sequences three x_0, x_1, x_2; so, x_0 is the past. So, the difference you note when we consider so, digital filter as if the input is coming from the left side and it is being shifted towards right. But here, the past is always shown at the in the figure in the top side and all the present are following successively. So, as you proceed in the time domain because it is unfolding over the time you know instances.

So, you are always the temporarily present sample or current sample will be shown at the bottom or at the layer, which is successive layers always the temporarily following inputs will be coming in the sequence. So, if there are say for example, the sequence length is 3. So, you can consider there are three such unfolding.

(Refer Slide Time: 10:12)



Unrolling of RNN

Usually drawn as:

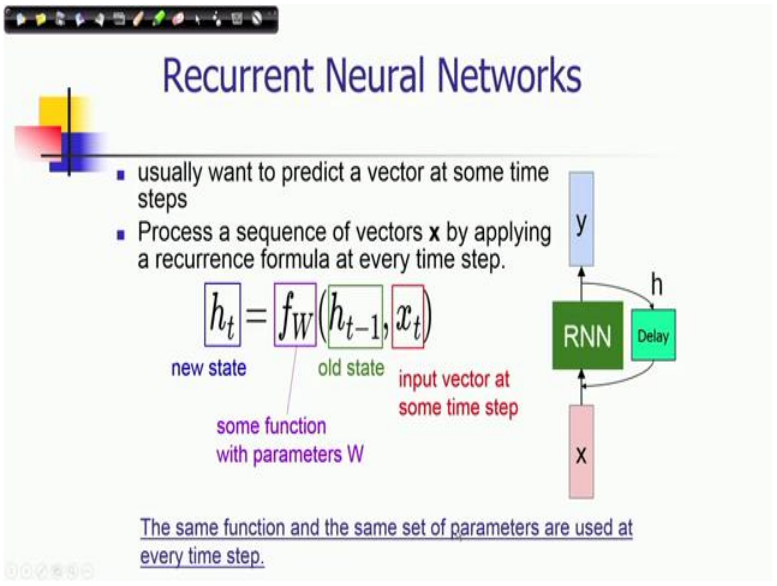
Learning algorithm: Back propagation through time (BPTT)

Courtesy: John Canny, UC Berkley.

You can draw it also in this fashion; that means, you provide input x_0 in the first stage get output y_0. Also it generates an intermediate output h_0, which could be equal to y_0 or which could be some function of y_0. And they that becomes the input to the next RNN, a next stage next block of neural block it is unfolding, which is the same RNN string, but at that time instead it is operating with input x_1 and also from the intermediate input h_0 generates y_1.

Next time stamp it will be input x_2, then h_1 both are input, that is intermediate output of the previous time instance is h_1 and it produces output y_2. Also generates an intimated output h_2, since there is no other input and we may not observe. So, maybe now your sequence has been generated as y_0, y_1, y_2 from x_0, x_1, x_2. That is a kind of explanation of the working of RNN. So, the learning algorithm what we use, if I unfold this the same back propagation algorithm can be used.

(Refer Slide Time: 11:24)



Recurrent Neural Networks

- usually want to predict a vector at some time steps
- Process a sequence of vectors **x** by applying a recurrence formula at every time step.

$$\boxed{h_t} = \boxed{f_W}(\boxed{h_{t-1}}, \boxed{x_t})$$

new state

old state

input vector at some time step

some function with parameters W

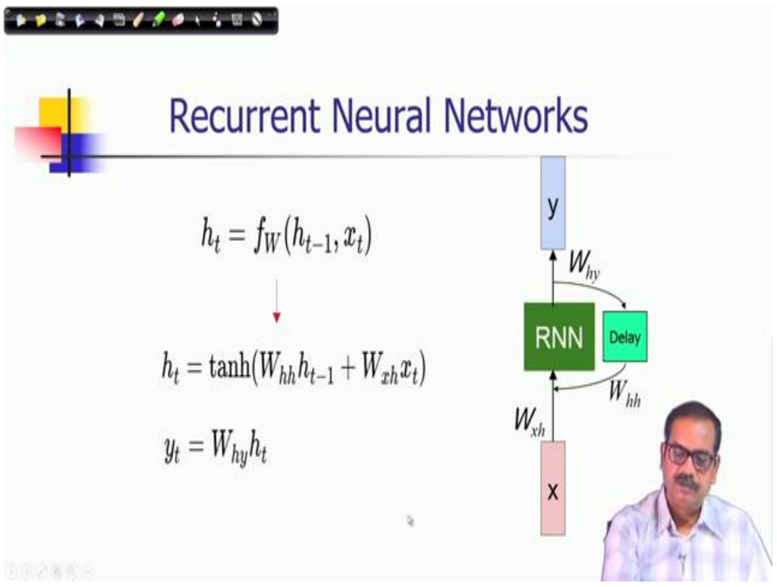
The same function and the same set of parameters are used at every time step.

y
h
RNN
Delay
x

So, this is trying to elaborate that particular aspect, that relationship between the intermediate output and also the past input or present input and past intermediate output. So, you can see that y is function of h_t and h_t is function of h_{t-1}, x_t of that RNN block where the w is the parameter.

So, h_{t-1} is the old state, x_t is the input victor at sometimes time and W is a function with parameter w and h_t is the new state. So, the same function and the same set of parameters are used at every time step. And y_t can be considered as a function of h_t it is a linear function, there is a W_{hy} is the weights connecting the intermediate output of the neuron to the final output.

(Refer Slide Time: 12:26)



Recurrent Neural Networks

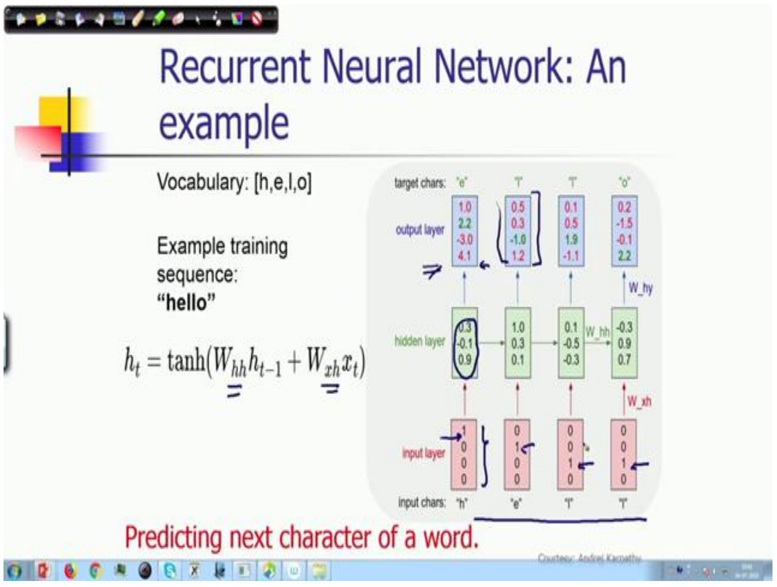
$$h_t = f_W(h_{t-1}, x_t)$$

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

$$y_t = W_{hy}h_t$$

So, one of the form of h_t could be that it is tan hyperbolic function aAnd it has been shown by this weight set. So, W_{hh} is a matrix, which is producing the vector h_{t-1}, which is producing this particular form and $W_{xh}x_t$ is another set of inputs, and relationship within y_t and h_t is it is $W_{hy}h_t$ linear combinations from h_t.

(Refer Slide Time: 12:57)



Recurrent Neural Network: An example

Vocabulary: [h,e,l,o]

Example training sequence: **"hello"**

$$h_t = \tanh(\underline{W_{hh}}h_{t-1} + \underline{W_{xh}}x_t)$$

Predicting next character of a word.

Courtesy: Andrej Karpathy

So, one particular example is shown here, say in this case it is a example has been drawn from the natural language processing or string processing. So, what we are trying to predict, we are trying to predict the next character of a word. So, there are training sequences.

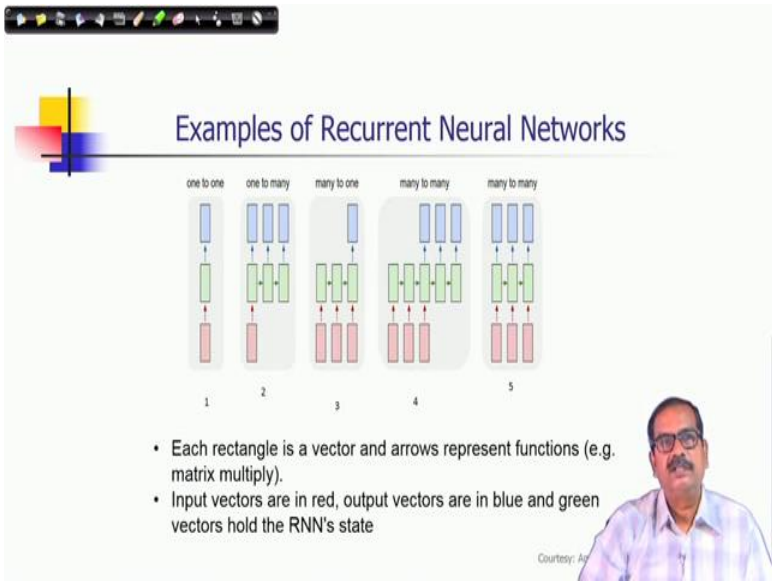
So, one may say hello is a training sequence and as you can see this RNN input. It can unfold the RNN into these seems there are the length of the string is five and there are prediction is four. So, we consider unfolding of RNN till fourth stages of input character and this is how the input is proceeded with temporal timestamp and this is the structure. And so, these are the weights at the stage and this is the weight for h_{t-1}, it is an one-dimensional input it is shown here.

Now, hidden layer has a three input sorry, hidden layer has a three node three vectors. So, accordingly you should have the weight vector W_{hh} and also $W_{xh}x_t$ that should come here and then you are using tan hyperbolic layer. So, there are three such nodes here and from there your output which is also considered as output of vector four dimensional vector.

Now, you are using here hot encoding; so, you can see the encoding which is used here. So, per h we are using a four dimensional hot encoding here. So, this position one shows h, this is e, this is l, this is l and your outputs which are being produced here yeah. The specified ground truths should have been here it should have been $\begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$, but instead you are producing this output. So, the last function will be determined in this fashion.

Similarly, this is a production of l, I mean it should have been l, but it produces this output and otherwise you know the ground truth should have been $\begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$. So, in this way the last function set to be defined.

(Refer Slide Time: 15:19)



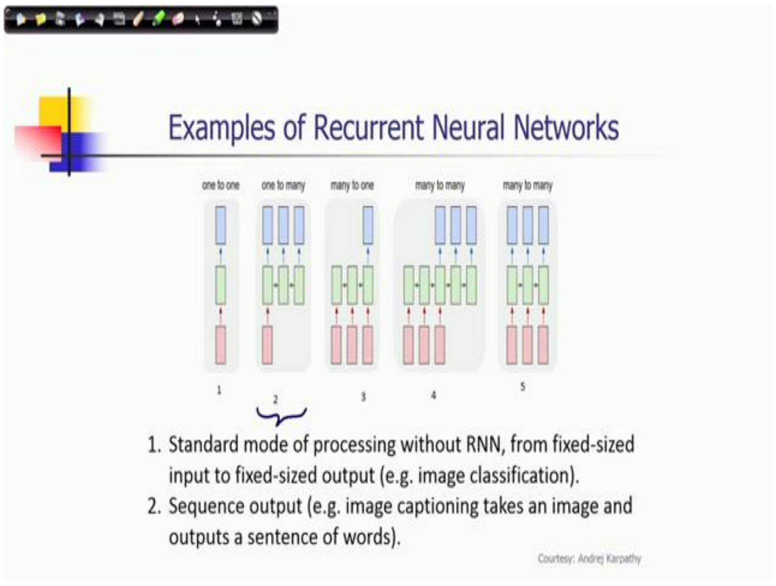
Examples of Recurrent Neural Networks

- Each rectangle is a vector and arrows represent functions (e.g. matrix multiply).
- Input vectors are in red, output vectors are in blue and green vectors hold the RNN's state

Courtesy: A

These are some examples, that what are possible configurations and how different kinds of neural networks could we use, different architectures could be used for solving different types of problem. Here, you can see that we have given some numbers and each rectangle is a vector and arrows represent functions and input vectors are shown in red and output vectors are in blue.

(Refer Slide Time: 15:47)



Examples of Recurrent Neural Networks

1. Standard mode of processing without RNN, from fixed-sized input to fixed-sized output (e.g. image classification).
2. Sequence output (e.g. image captioning takes an image and outputs a sentence of words).

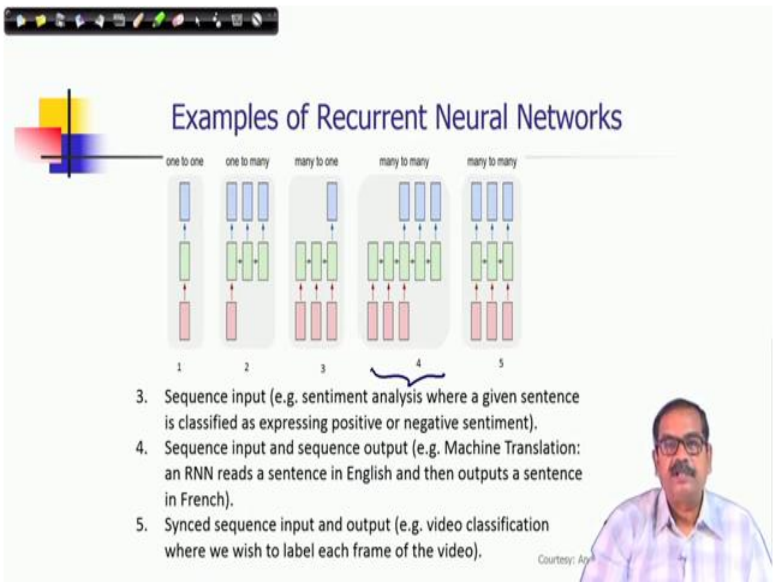
Courtesy: Andrej Karpathy

So, let me just explain. So, suppose these are different configurations first one is one to one configuration which means input is a fixed vector and output is also fixed size output.

So, example is an image classification and this is a conventional neural network processing, you do not require RNN for this kind of processing. Where as a second one which means this particular structure the input is a fixed size input or output is a sequence.

Now, here you require a RNN and so, it is a second of output the example could be the image captioning, which takes an image and outputs the sentence of word.

(Refer Slide Time: 16:38)



Examples of Recurrent Neural Networks

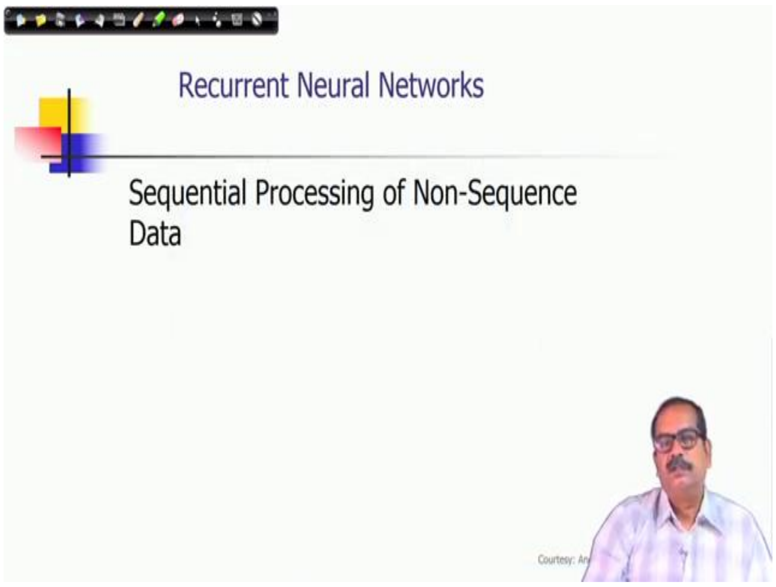
3. Sequence input (e.g. sentiment analysis where a given sentence is classified as expressing positive or negative sentiment).
4. Sequence input and sequence output (e.g. Machine Translation: an RNN reads a sentence in English and then outputs a sentence in French).
5. Synced sequence input and output (e.g. video classification where we wish to label each frame of the video).

Courtesy: A

The third one is, so this is a sequence output is fixed. So, typical example could be that say sentiment analysis, where a given sentences classified as expressing positive or negative statement sentiment. Fourth applications, so these are fourth one where input is a sequence also output is a sequence and this is an example could be the machine translation. It reads a sentence in English and then outputs the sentence in French for example, a typical example of a problem, here also RNN should be use.

So, only for the first one it is a conventional network for all these kinds of sequence processing, we need to use RNN. Consider the fifth one you have the synchronous, no sequence input output sequence and it could be the video classification for labelling each frame of the video.

(Refer Slide Time: 17:46)



Recurrent Neural Networks

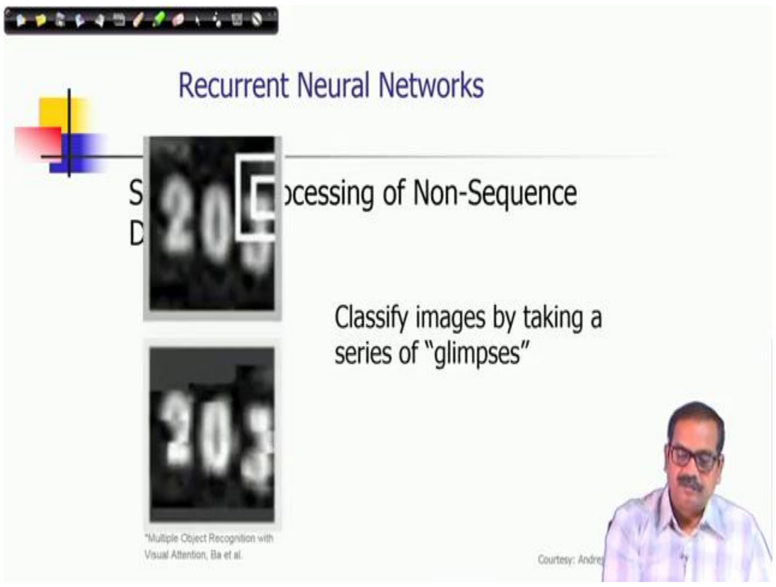
Sequential Processing of Non-Sequence Data

Courtesy: An

So, these are some examples of RNN, use of RNN in different applications. Sometimes you can convert even a pixel input as a sequence by following certain predetermined order of scanning the input data.

So, we call it sequential processing of non-sequence data; for example, for image we can scan from left to right and top to bottom. Now that itself will provide you some sequence of regions blocks. Now, those could be input to RNN and we can process it for various purposes.

(Refer Slide Time: 18:20)



Recurrent Neural Networks

S ocessing of Non-Sequence D

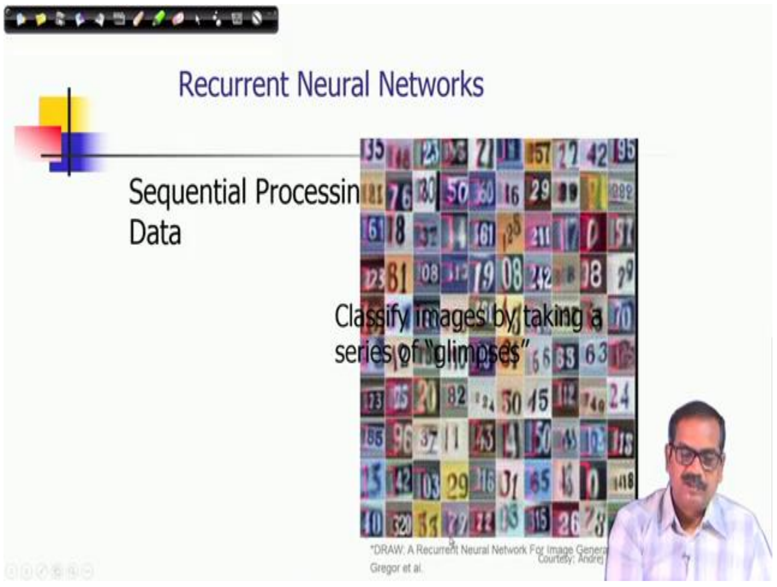
Classify images by taking a series of "glimpses"

*Multiple Object Recognition with Visual Attention, Ba et al.

Courtesy: Andre

So, this is just this is showing a one kind of processing. So, the blocks are moving in a sequence and we would like to classify images by taking a series of glimpses.

(Refer Slide Time: 18:33)



Recurrent Neural Networks

Sequential Processin Data

Classify images by taking a series of "glimpses"

*DRAW: A Recurrent Neural Network For Image Genera Gregor et al.

Courtesy: Andrej

Another example classify images by taking series of glimpses

(Refer Slide Time: 18:38)

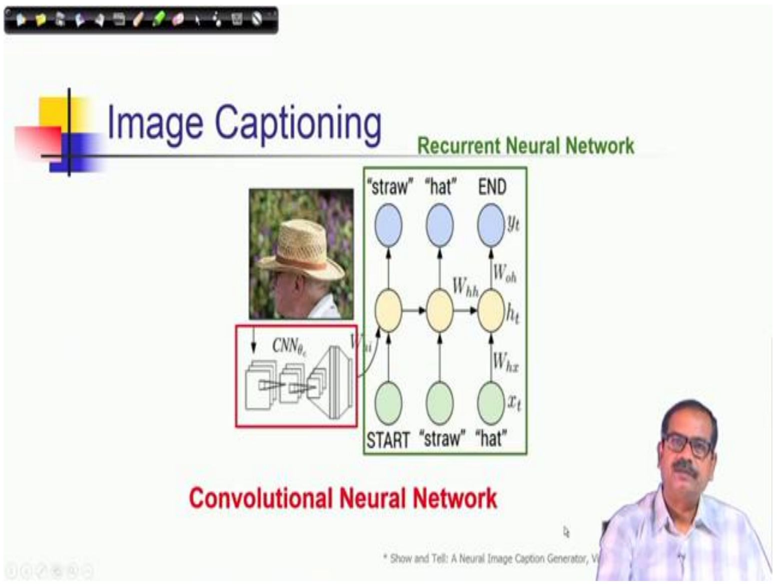


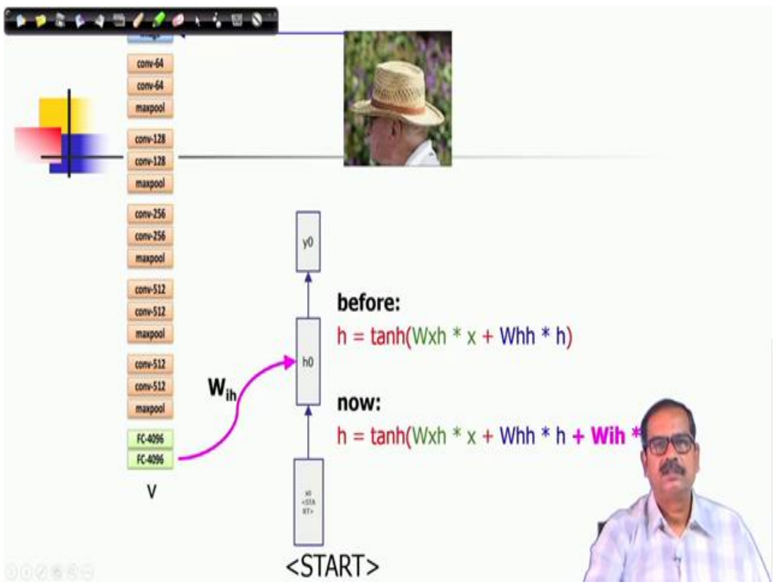
Image Captioning

* Show and Tell: A Neural Image Caption Generator, V

So, the application of this kind of one such very popular application is image captioning, where image regions we also can scan. But in this case what it does that it represents the image as a feature vector using a CNN and that is input to an RNN and the descriptions one by one the descriptions of the output. Once again the number of the length of the output sequence depending we decide the length of the unfolding of RNN that has to be there.

So, for example, in this particular figure the image could be described as the straw hat and that is the sentence; so, it has to start and it has an end. So, with that kind of specifications you are training the RNN in this way. So, you use the convolution neural network to describe the feature and then, followed by recurrent neural network to extract the caption.

(Refer Slide Time: 19:42)

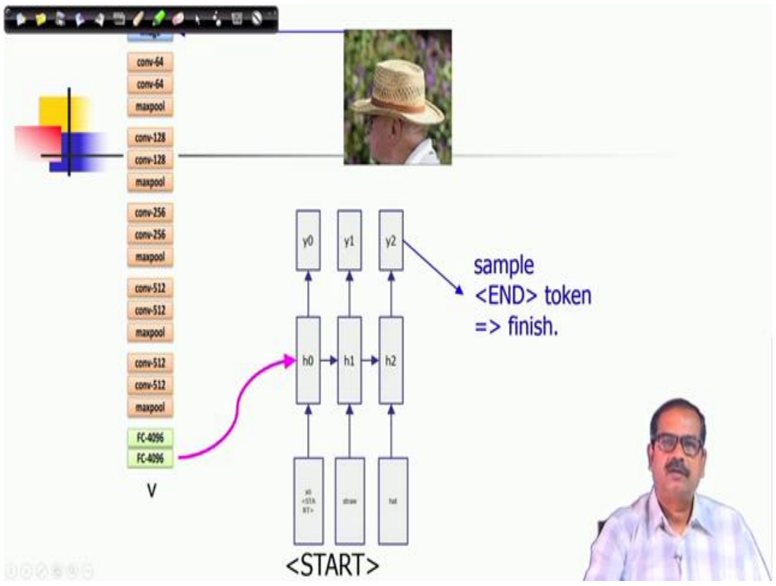


So, it just shows the workflow that you use a convolution neural network, where image is an input and it goes through several layers of convolution neural network and finally, you represent it is a feature vector. In fact, the last layers of fully connected layers they are mostly used for classification, you can remove that layer and you can use only the representation of the last layer of the fully connected layer FCporchiporchi

4096; that means, the feature dimension is 4096 at the last layer in this case.

So, that could be the input to your RNN and also the input another input is a start symbol from where you are starting the sentence reconstruction, it reconstructed the sample why not, that itself is the input towards your intermediate sample.

(Refer Slide Time: 20:39)



So, these are sample and it goes like this. So, in the same (Refer Time: 20:44) way RNN operates and then it can generates a sentence. So, as you train on a number of images it depending upon the image content it tries to predict those levels which are being trained in a network.

(Refer Slide Time: 20:59)

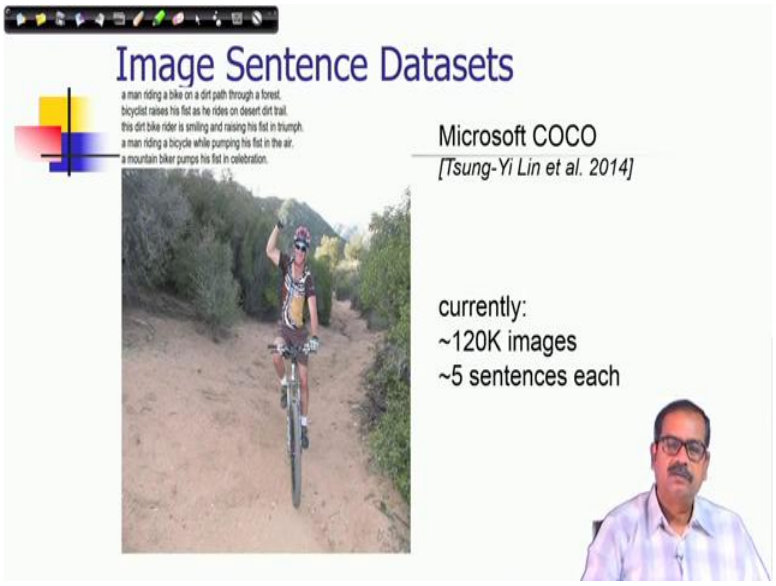


Image Sentence Datasets

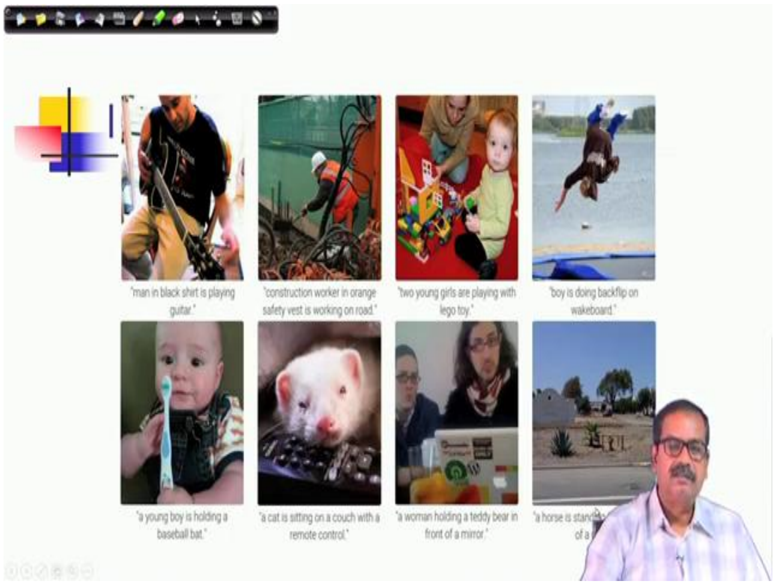
a man riding a bike on a dirt path through a forest.
bicyclist raises his fist as he rides on desert dirt trail.
this dirt bike rider is smiling and raising his fist in triumph.
a man riding a bicycle while pumping his fist in the air.
a mountain biker pumps his fist in celebration.

Microsoft COCO
[Tsung-Yi Lin et al. 2014]

currently:
~120K images
~5 sentences each

There is a data set which is called Microsoft COCO and it has say 120 K images and for each image there are 5 sentences and used it for training and there is an example. So, you provide this as training set to the network and then you can generate different kind of image captions.

(Refer Slide Time: 21:18)



"man in black shirt is playing guitar."

"construction worker in orange safety vest is working on road."

"two young girls are playing with lego toy."

"boy is doing backflip on wakeboard."

"a young boy is holding a baseball bat."

"a cat is sitting on a couch with a remote control."

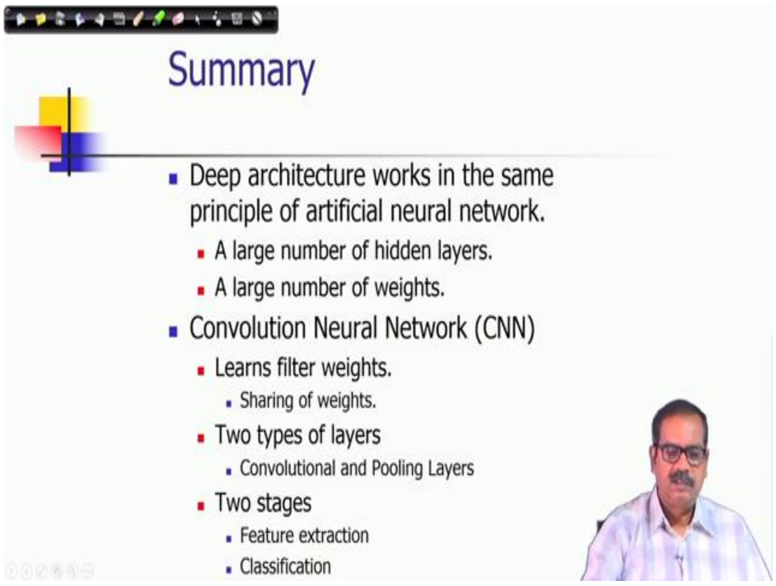
"a woman holding a teddy bear in front of a mirror."

"a horse is stand

of a

So, some examples are shown here consider the first row top row and the leftmost image. Where, you have the man in black shirt is playing guitar and a say other one is shown as construction worker in orange safety (Refer Time: 21:39) he is working on road. So, these are a very interesting results and presently a lot of work is going on video summarization, image captioning and many other things using this kind of network.

(Refer Slide Time: 21:55)



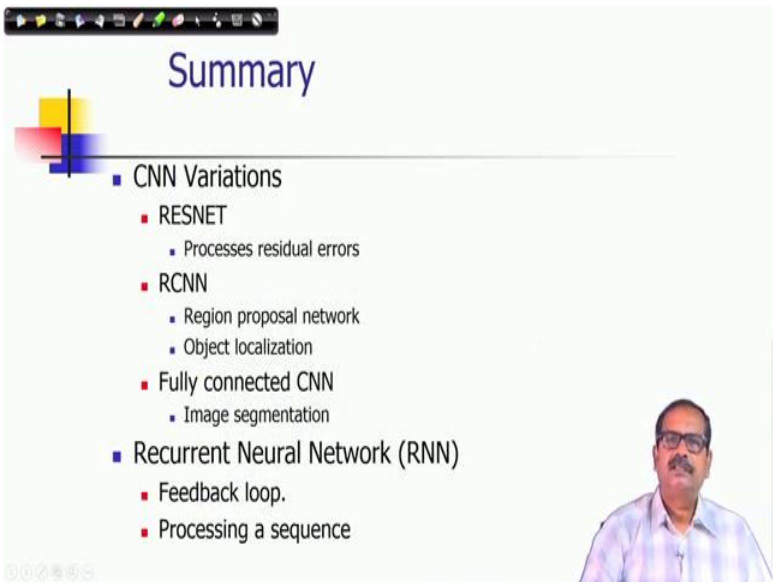
Summary

- Deep architecture works in the same principle of artificial neural network.
  - A large number of hidden layers.
  - A large number of weights.
- Convolution Neural Network (CNN)
  - Learns filter weights.
    - Sharing of weights.
  - Two types of layers
    - Convolutional and Pooling Layers
  - Two stages
    - Feature extraction
    - Classification

So, let me summarize the topics and whatever we have covered under this deep neural architecture and its applications. So, we have seen that deep architecture works in the same principle of artificial neural network, but it has a large number of hidden layers and it has a large number of weights.

One a very popular deep neural architecture is a convolution neural network, because it can learn filter weights and it can share those weights and there are two types of layers usually in this process. One is convolution layer another is this is pooling layers for the feature description, and there are two stages which are involved feature extraction and classification. So, in the classification stage usually we used fully connected neural networks.

(Refer Slide Time: 22:50)



Summary

- CNN Variations
  - RESNET
    - Processes residual errors
  - RCNN
    - Region proposal network
    - Object localization
  - Fully connected CNN
    - Image segmentation
- Recurrent Neural Network (RNN)
  - Feedback loop.
  - Processing a sequence

So, there are variations of convolution neural networks say RESNET it processes residual errors then RCNN it has a region proposal network and it localizes object and also it could be fully connected CNN where, it uses image segmentation. And then there is recurrent neural network where it is a different kinds of neural architecture, it is quite different from convolution neural network, because it has a feedback loop it is not a feed forward network.

And recurrent neural networks are mainly used for processing a sequence. So, with this let me stop here. And this is the end of this particular topic and this lecture.

Thank you very much for listening to my lecture.