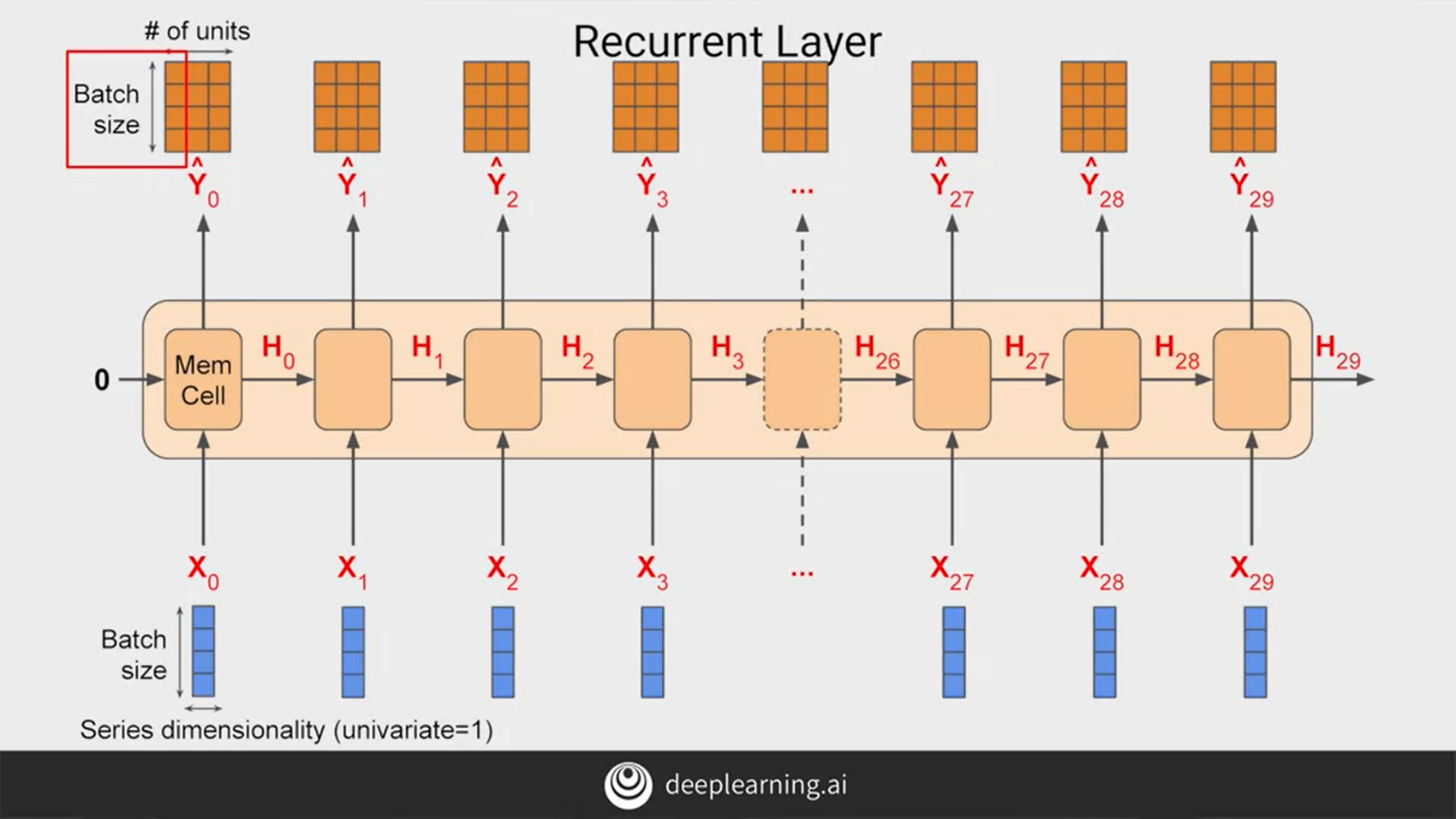
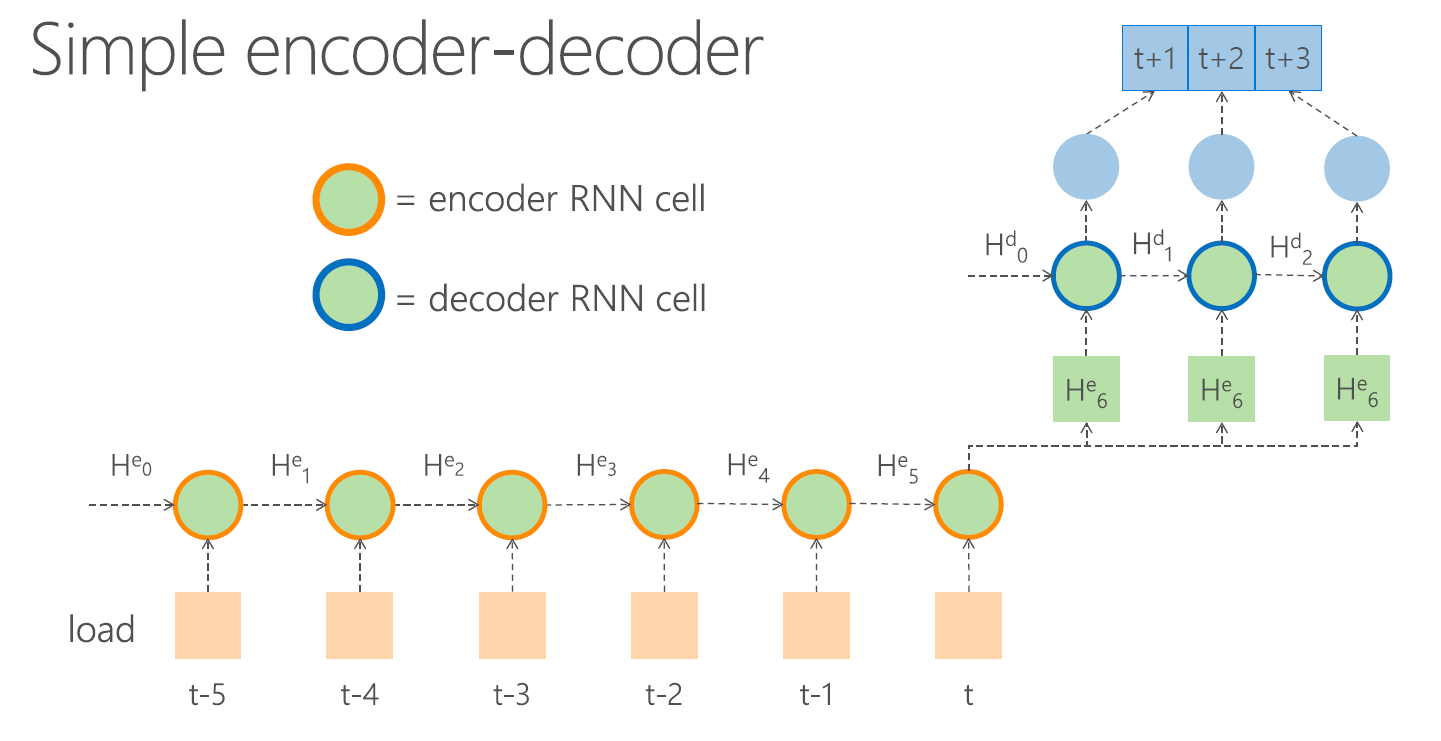
# Understanding LSTMS

## Requirements

* Our window size for time series is 30
* Our batch size is 4
* Input dimension is 1
* So in the picture below, our input is 4 x 1 (at each timestep with 30 timesteps)
* At the fist timestep the hidden state is 0
* Hidden unit of Lstm is 3
* The first output: (4 x 1) x (1 x 3) = 4 x 3, this becomes H0 which is the same as Y0
* The second output: (4 x 1) | (4 x 3) concatenated across column axis to give (4 x 4) which is then multiplied by the hidden state of 3 units, so (4 x 4) x (4 x 3) = 4 x 3
* The third output Y2: (4 x 1) | (4 x 3) concatenated across column axis to give (4 x 4) which is then multiplied by the hidden state of 3 units, so (4 x 4) x (4 x 3) = 4 x 3 and this step keeps repeating itself for 30 timesteps, so the output of the entire layer is the next point.
* There are 30 timesteps, So the output of the **full layer** is 4 x 30 x 3

## 1.2 Using the Time Distributed Layer

* Without using the Time Distributed layer and just having a dense layer at the end. We are only outputting the predicted values at the **last time step** (Return Sequences=False).
* By using a Time Distribued layer, we output the predicted values at each time step.
* Return Sequences=True, is for outputting all of the values at each time step if we are adding another layer to the Recurrent Neural Network, same as the Time Distributed, except that we are adding another recurrent layer.



## 1.3 Using the Auto Encoder/Repeat Vector

* Looking at the image below, we can see that we have a repeatvector(3), after the last time step, a predicted value is output 3 times, i.e. H6.
* So a sequence to vector. Next we add another RNN layer, Return Sequences=True means that we output all of the values at each time step in H6 to the next RNN layer.
* Flatten, Just combines the 3 time steps into one vector

