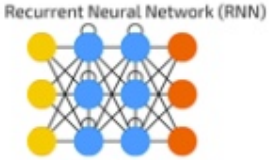


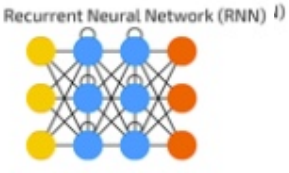
An Introduction To Recurrent Neural Network

Spotle.ai

Instructor

Kaniska Mandal
Sr. Principal Bigdata Machine Learning
Enlighted Inc
California

MS in CS with specialization in Machine Learning
Georgia Institute of Tech

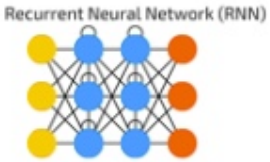


Introduction

Dear Friends. My name is Kaniska Mandal.

I had the privilege to solve practical business problems using machine learning algorithms and data science techniques in the domain of eCommerce, Cloud operations, Healthcare, Smart Networking and Industrial IOT, Smart Traffic as part of different startups like Kitenga, Tesseract, Enlighted, Sensity Systems and large enterprise like Dell RnD, Walmart Labs.

I would like to share some of the great stuff that I could learn over the years. Today I am going to focus on a specific machine learning algorithm - RNN or Recurrent Neural Network.

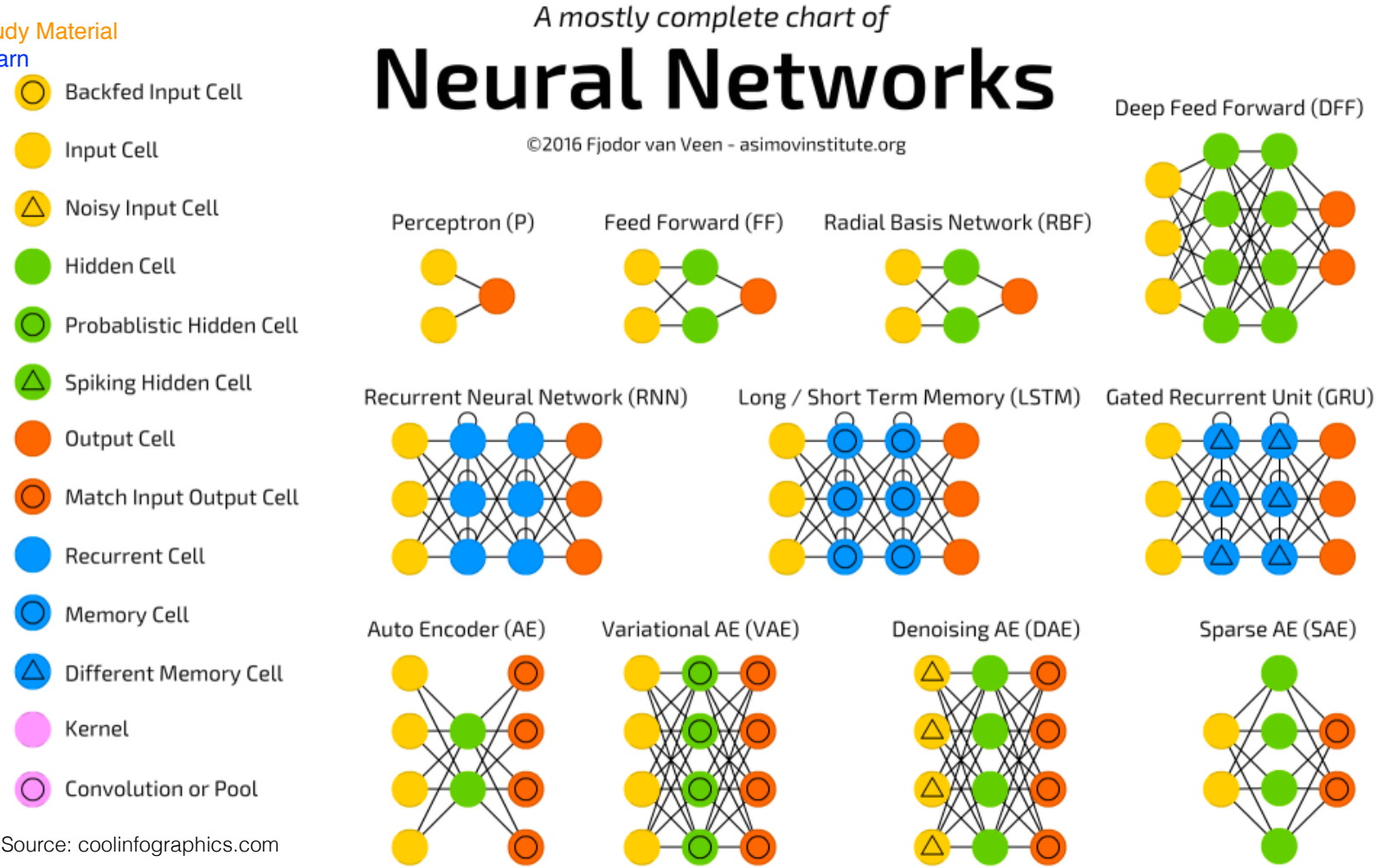


The World of Neural Networks

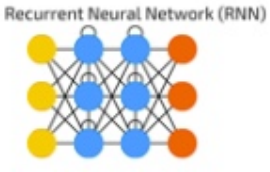
Spotle.ai

You all definitely have identified exciting patterns in nature at some point of time in your life.
Its fascinating to see that there is a hidden chain of algorithms with certain mathematical rules, governing everything around us. Inspired by the intricate rules of how human brain works, computer scientists have built a great set of neural network models through curious observations and mathematical studies over the centuries.

Spotle.ai Study Material
Spotle.ai/Learn



Source: coolinfographics.com



What is RNN

Today, we shall focus on a specific type of problems which demonstrates sequential patterns in dataset. For example, paragraphs are sequences of sentences, sentences are sequences of words and words are sequences of characters. As I speak, continuous stream of audio and video frames are being generated over time. Recurrent Neural Networks are best models to solve such sequential problems.

Lets consider this sentence “I had delivered my package”, now lets change the order of words as “I had my package delivered”. So now the sentence means - I took help of someone else to deliver the package.

Lets try to pronounce “wreck a nice beach” bit faster.

How does it sound like ? “recognize speech”.

Definitely we are taking about a beach not the speech.

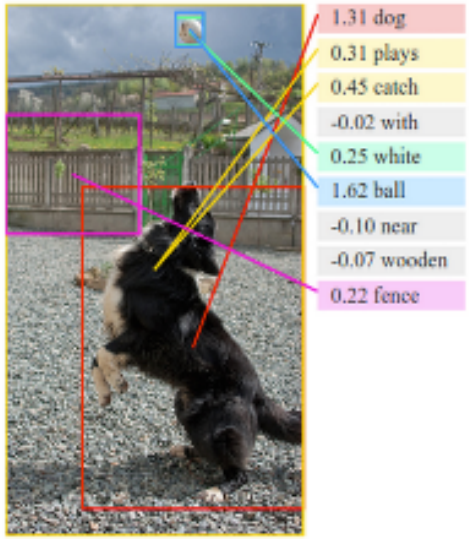
In both the cases we need to remember the insights gained from prior analysis of the words. If we lose the context we won’t be able to reconstruct the meaning.

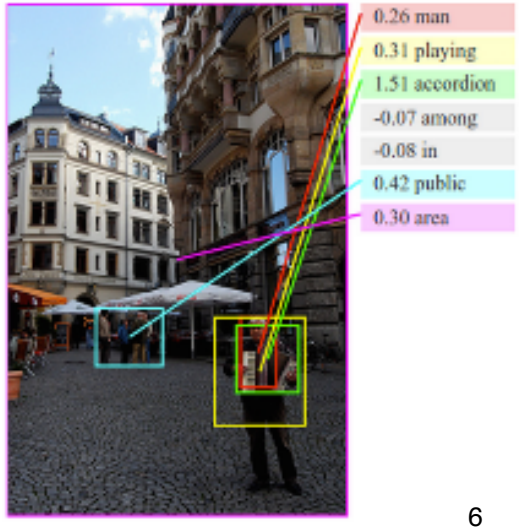
RNN remembers the context along the time line by doing the same operation over and over. Hence called Recurrent.

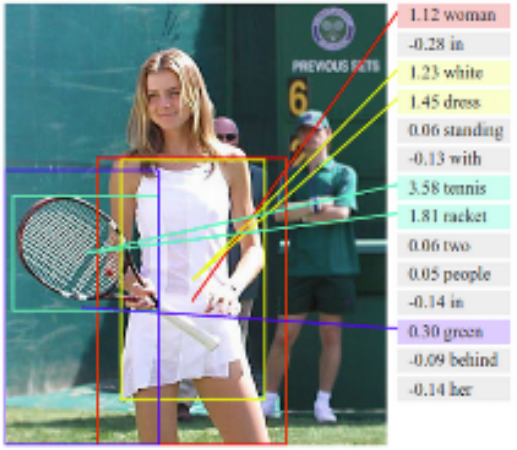
Applications of RNN

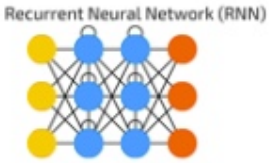
RNN models are applied to different types of problems like Sequence Generation, Sequence Labeling, Sequence Classification and Sequence Translation.

1. Given a sequence of words, find the probability of occurrent of the next word.
2. Translate text from one language to another language.
3. Find the likelihood of phonetic segments based on input sound waves and formulate a word.
4. Find the sequence of events inside an image and annotate the events.
5. Describe the images inside a video frame by frame using sequential tagging.





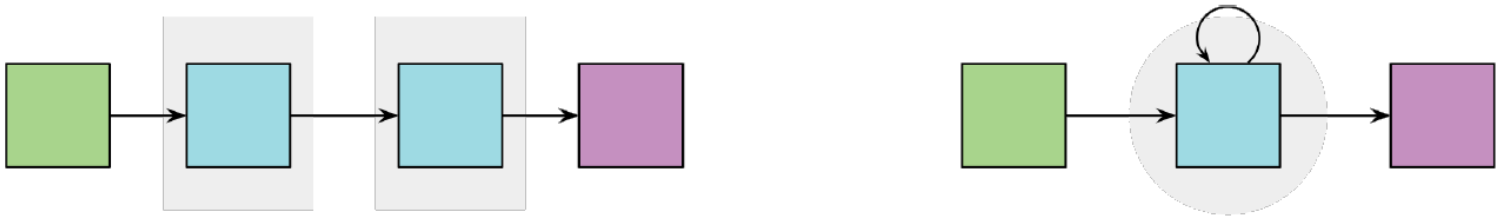




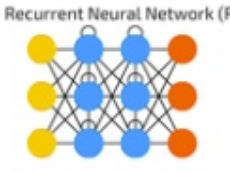
RNN - Peeling the Layers

Just like the simple Feed Forward Neural Network, RNN sums up weighted inputs from other neurons while working on current neurons.

But the main difference is, RNN allows neurons to connect in the forward direction to higher layers and also in the backward direction to lower layers, thereby forming cycles.

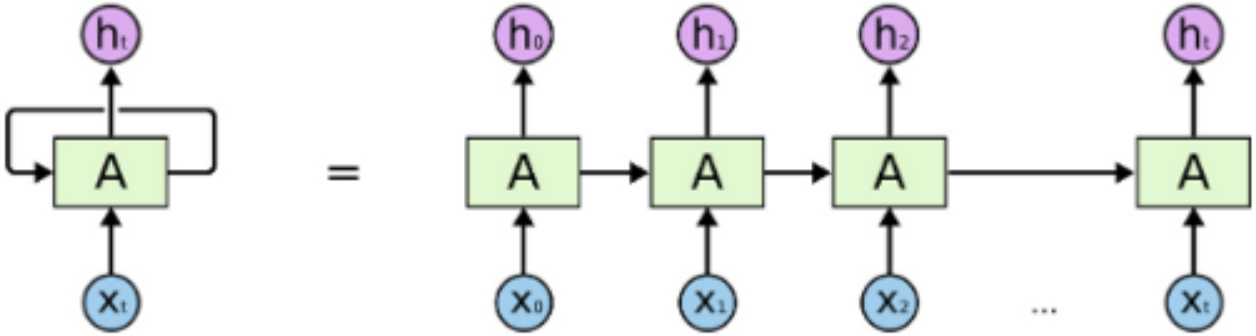


RNN allows us to remember the hidden network as state which has access to all of the past inputs of the sequence and it acts as a working memory.



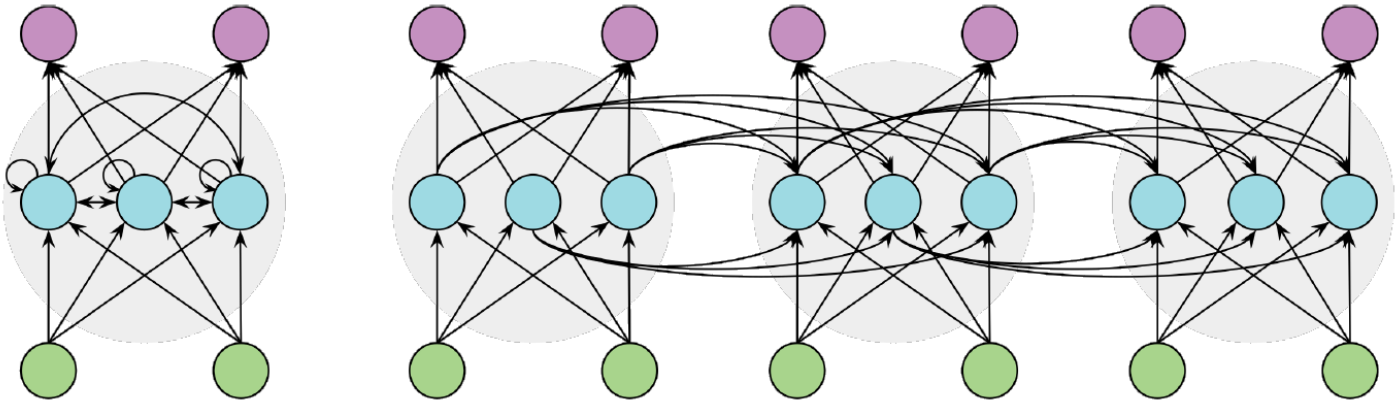
RNN - Peeling the Layers

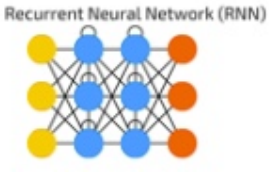
RNN allows us to remember the hidden network as state which has access to all of the past inputs of the sequence and it acts as a working memory.



Spotle.ai Study Material
Spotle.ai/Learn

Overall, this is how it looks like when all the cells are combined.





RNN
Mathematical Representation

Spotle.ai

For the sake of understanding the basics of RNN

Consider the following representation of a recurrent neural network:

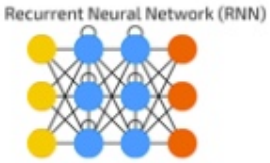
$$\mathbf{h}_t = \sigma(\mathbf{U}\mathbf{x}_t + \mathbf{V}\mathbf{h}_{t-1})$$

Where ***U*** and ***V*** are the weight matrices connecting the inputs and the recurrent outputs respectively. We then often will perform a softmax of all the $\mathbf{h}_t$ outputs (if we have some sort of many-to-many or one-to-many configuration). Notice, however, that if we go back three time steps in our recurrent neural network, we have the following:

$$\mathbf{h}_t = \sigma(\mathbf{U}\mathbf{x}_t + \mathbf{V}(\sigma(\mathbf{U}\mathbf{x}_{t-1} + \mathbf{V}(\sigma(\mathbf{U}\mathbf{x}_{t-2})))$$

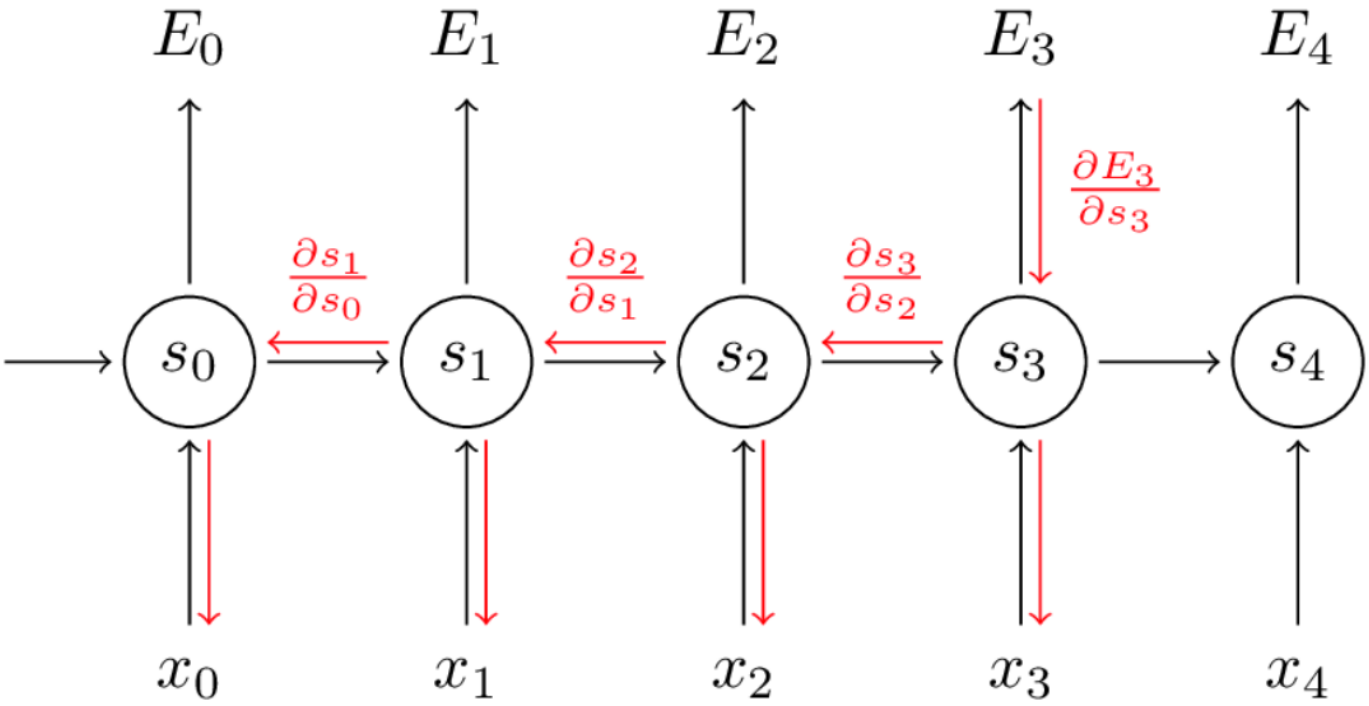
From the above you can see, as we work our way back in time, we are essentially adding deeper and deeper layers to our network.

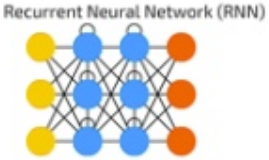
$$\frac{\partial E_3}{\partial U} = \frac{\partial E_3}{\partial out_3}\frac{\partial out_3}{\partial h_3}\frac{\partial h_3}{\partial h_2}\frac{\partial h_2}{\partial h_1}\frac{\partial h_1}{\partial U}$$



RNN - Unfold in Timeline

So we have just seen how RNN can be unfolded in time and the gradient of the errors can be back propagated. This process is called Back-Propagation Through Time (BPTT).

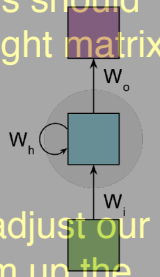




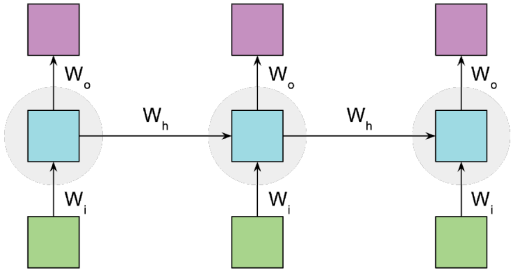
RNN - Unfold in Timeline

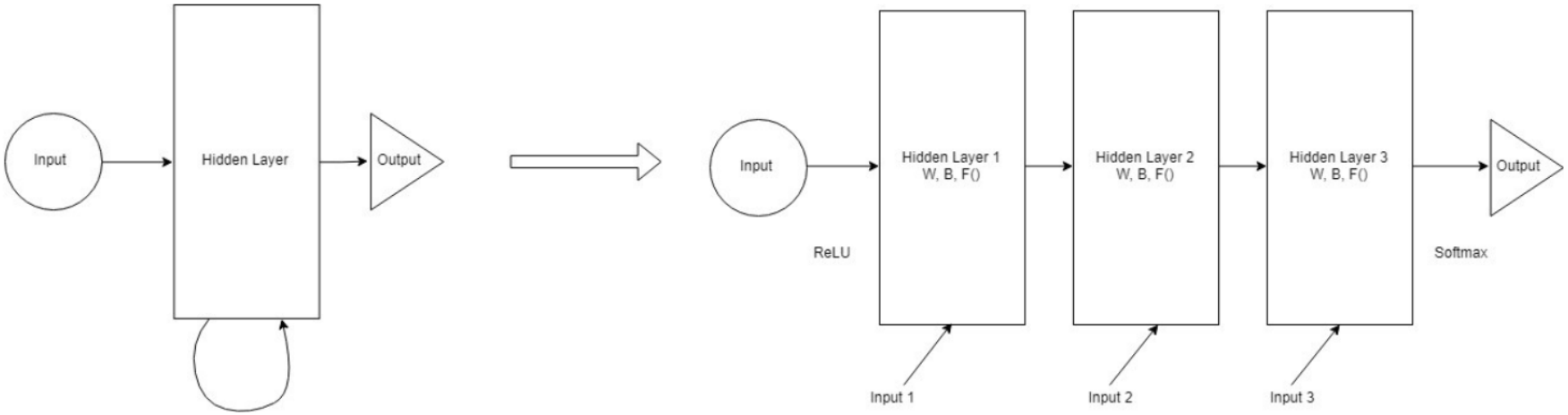
Spotle.ai

Backpropagation refers to "weighted average delta" which specifies "the direction and amount we want the value of this neuron to change next time. It amplifies the error so that we know by how much each layer's nodes should move up or down. Once we know this, we can just update each weight matrix.

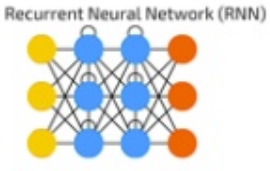


For each weight, we multiply its output delta by its input value and adjust our weight by that much (up or down). The key difference is that we sum up the gradients for W at each time step. In a traditional NN we don't share parameters across layers, so we don't need to sum anything. So BPTT is standard backpropagation on an RNN unfolded over the timeline.





Unrolling a RNN

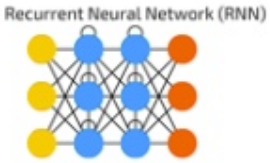


RNN - Issues

Vanishing Gradient Problem
Sometimes simple RNN can not interpret long-distance relationships i.e. words which are several time-steps apart.

“The man who wore a wig on his head went inside”. A simple RNN may consider that a ‘wig went inside”. But the sentence is really about a man going inside, not about the wig. We won’t go into much details but key take away is that we can improve the RNN by using a sophisticated activation function like ReLU (Rectified Linear Unit) in order to avoid vanishing gradients.

Overfitting
Since RNN shares error between all the weights, if a particular configuration of weights accidentally generates perfect correlation between the output dataset and prediction, then neural network will stop learning and it will not provide heaviest weights to its best inputs any longer. Thereby it will miss many important insights.

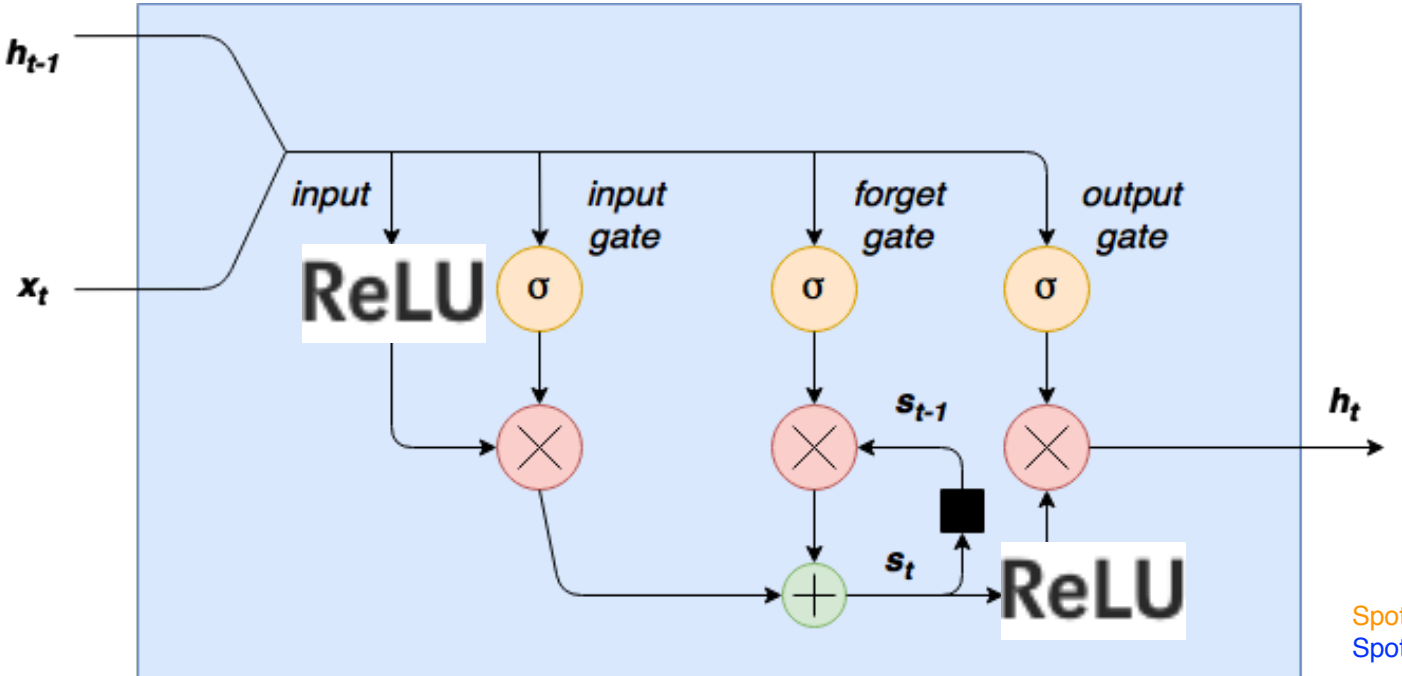


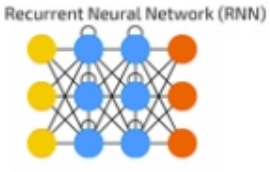
LSTM - Solution to issues of Simple RNN

At the heart of an RNN is a layer made of memory cells. The most popular cell at the moment is the Long Short-Term Memory (LSTM) which maintains a cell state as well as a carry for ensuring that the signal (information in the form of a gradient) is not lost as the sequence is processed. At each time step the LSTM considers the current word, the carry, and the cell state.

The LSTM has 3 different gates and weight vectors: there is a “forget” gate for discarding irrelevant information; an “input” gate for handling the current input, and an “output” gate for producing predictions at each time step.

The function of each cell element is ultimately decided by the parameters (weights) which are learned during training.

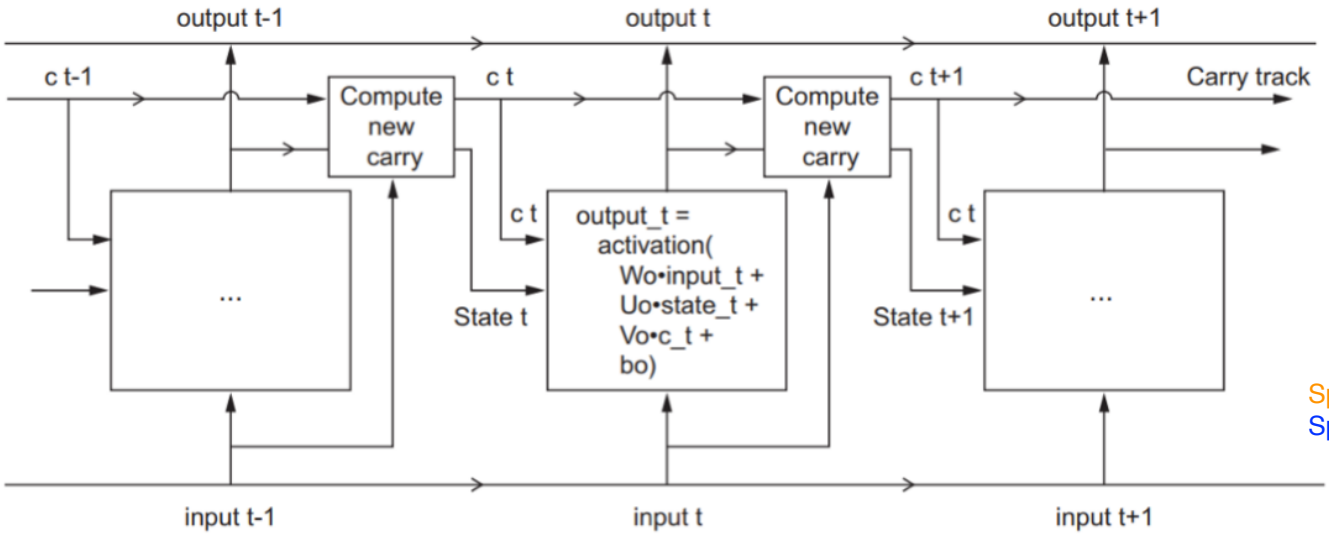




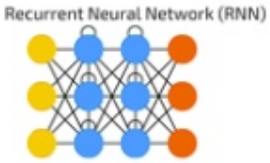
RNN-LSTM Implementation Workflow

Spotle.ai

- Lets formulate the workflow of training an RNN to auto-generate abstract of a paper.
- Create Features and Labels from sequences.
  - Sequence of words mapped to integers and converted to vectors using embedding matrix. An Embedding maps each input word to a 100-dimensional vector. The embedding can use pre-trained weights.
- Build RNN model with embeddings, LSTM and Dense Layers.
  - Pass the pre-trained embeddings to LSTM layer.
  - LSTM uses dropout to prevent overfitting.
  - LSTM may use a Masking layer to mask any words that do not have a pre-trained embedding which will be represented as all zeros.
  - LSTM uses a fully-connected Dense layer with ReLU activation.
  - LSTM uses Adam optimizer (BPTT) and trained using Categorical CrossEntropy loss. The gradients of the parameters are calculated using back-propagation and updated with the optimizer. During training, the network will minimize the log loss by adjusting the pre-trained weights or trainable parameters (weights).
- Train the model and make predictions.
  - We pass in a starting sequence of words, make a prediction for the next word, update the input sequence, make another prediction, add the word to the sequence and continue for however many words we want to generate.



Spotle.ai Study Material
Spotle.ai/Learn



RNN - Summary

At the end of this quick overview, we can summarize the main concept of RNN as follows

— The context and meaning of a sequence can be preserved through advanced Back Propagation Through Time algorithm and we can derive correct insights using LSTM model.