[**11.1.1: Setting the context**](#_1pd3d4v84ib0) **1**

[**11.1.2: Introduction to sequence learning problems**](#_7qalz87ye5l5) **3**

[**11.1.3: Some more examples of sequence learning problems**](#_eq8s4ay9gzy2) **4**

[**11.1.4: Sequence learning problems using video and speech data**](#_xkheoldqiqg2) **5**

[**11.1.5: A wishlist for modelling sequence learning problems**](#_wp04s8uy5k1s) **6**

[**11.1.7: Intuition behind RNNs**](#_k6nm4nxs0wy7) **7**

[11.1.7a: Part 1](#_15483qpppld2) 7

[11.1.7b: Part 2](#_3f05k8569jol) 8

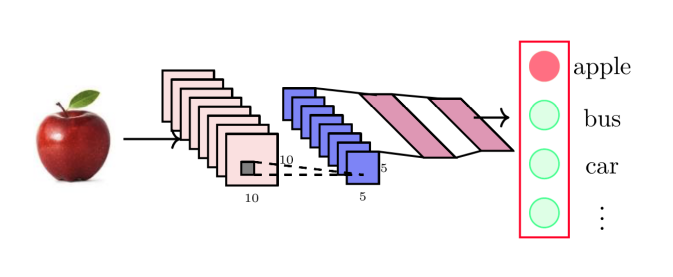
[**11.1.8: Introducing RNNs**](#_q8k9tqmdkum) **9**

[**11.1.9: Summary and what next**](#_jt9v1bggf40f) **10**

**11.1: Sequence Learning Problems**

## 11.1.1: Setting the context

What kind of networks and inputs have we seen so far?

1. Firstly, we learned about Fully connected networks, where every neuron in a layer is connected to every neuron in the next layer.
2. An example would be using biological parameters as predictors for health risks 
3. The next set of networks we saw were Convolutional Neural Networks, where we saw concepts such as sparse connectivity and weight sharing.
4. An example would be using a CNN to classify the object in an image
5. In both Fully Connected networks and Convolutional Networks, the following points hold true
   1. **Outputs are independent of previous inputs:** The values of earlier inputs have no bearing on the output value corresponding to subsequent inputs.
   2. **Input is of a fixed length:** All inputs provided are of the same length/dimensions i.e no. of features / image dimensions are same
6. While both of these points were intuitively understood, we have explicitly mentioned them now as they are relevant to the next type of network that we are about to look at

## 11.1.2: Introduction to sequence learning problems

Let’s look at a different class of problems.

1. Consider the task of predictive texting. Let’s look at an example where our task would be to look at the current input character and predict the next character.
   1. Here, let us consider the prediction of the word “deep”
   2. Characters are given as a One-hot representation of a 26 dimension vector.
   3. When “d” is given as the input, the model predicts “e” as the most likely letter to follow.
   4. Now, when “e” is given as the next input, the model factors it in as well as the previous input “d” and gives us the prediction “e”.
   5. Now, we follow it with another input of “e”. The model considers the previous two inputs of “d” and “e”. It is then intuitive for it to predict “p”
   6. Once ‘p” is provided as the final input, combining it with the previous 3 inputs, we have “deep”, thus prompting the model to end the word.
2. Thus the following properties are exemplified by the network:
   1. **Outputs depend on previous inputs as well:** Values of previous inputs govern how current output is predicted
   2. **The length of the input is not fixed:** We saw an example for a 4-character word like deep, however it applies for any length of input.
3. The same principle applies even in sentence completion, where the model factors in the last 3 to 4 words typed to predict the next word in the sequence.

## 

## 11.1.3: Some more examples of sequence learning problems

What about a sequence of words?

1. Let’s look at another common problem known as Part of Speech (POS) tagging
   1. The model’s confidence in predicting that “movie” is a noun will be much higher if the previous input “awesome” was predicted as an adjective
2. Input words are given as One-Hot representations
3. However, do we always need to produce an output at every time step?
4. In problems involving “Sentiment Analysis”, we analyse the input words of a sentence and then determine if the sentiment is positive or negative. 
   1. This is a type of sequence classification problem, where we want to assign a class to an entire sequence of inputs rather than assigning a class to each input in the sequence as with POS tagging.

## 

## 11.1.4: Sequence learning problems using video and speech data

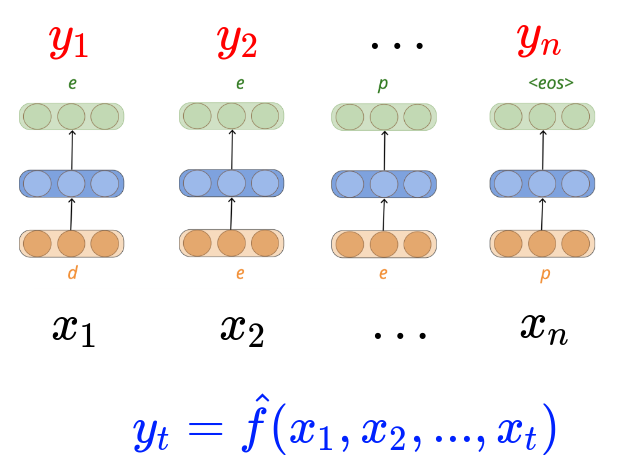
What are some other types of sequences that we encounter?

1. Speech can be considered as a sequence of phonemes just as how a sentence is a sequence of words. 
2. Video can also be considered as a sequence of frames. The following example involves analysing video footage of the surya namaskar asana being performed.

## 

## 11.1.5: A wishlist for modelling sequence learning problems

How do you model such sequence learning problems?

1. Let’s look at the problem of classifying each input in the sequence
   1. We need to establish the following 3 conditions for modelling sequence learning problems
      1. Ensure that ***yt*** is dependent on previous inputs also
      2. Ensure that the function is able to deal with variable number of inputs.
      3. Ensure that the function executed at each time step is the same
2. What about in the case where we need one output for the entire sequence of inputs?
   1. We need to establish the following 2 conditions
      1. Ensure that ***yt*** is dependent on previous inputs also
      2. Ensure that the function is able to deal with variable number of inputs.

## 

## 11.1.6: Intuition behind RNNs

### 11.1.6a: Part 1

What is the function being executed at each time step?

1. Let us look at the function that is executed at every time step
   1. Consider a
   2. In both of these cases, the parameters W2 and b2 remain the same
   3. This is applicable for every time step.
   4. Thus by **Parameter Sharing** for every timestep, we ensure that the same function is executed each time.
2. Now, let’s rewrite the equations using notation more relevant to sequence inputs.
   * 1. This is the equivalent term to hi
     2. s is used to denote “state”
     3. U is used to represent W1
     4. b is used to represent b1
     5. V is used to represent W2
     6. c is used to represent b2
3. Now, we have satisfied the point of ensuring that the same function is applied for every time step, thereby also satisfying the point of ensuring that the function can deal with variable number of inputs (The function works for x1, 2, … n Where n can be any non-zero +ve integer)
4. For example, yi depends on si which in turn depends on xi and not all previous inputs.

### 

### 11.1.6b: Part 2

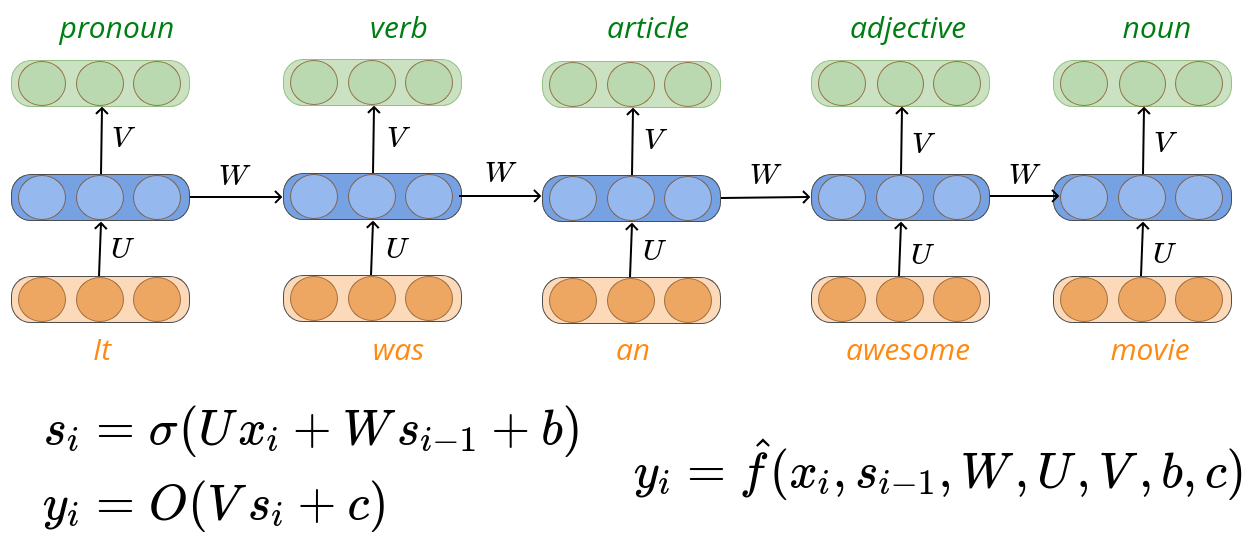
How does parameter sharing help?

1. Though we have satisfied the points involving function homogeneity and input variability, we are yet to satisfy the point that ensures dependency on previous inputs.
2. yi depends on si which in turn depends on xi and not all previous inputs.
3. What have we not accounted for so far?
4. Let’s try concatenating the inputs for every time step.
   1. Here, we have now satisfied dependency on previous inputs
   2. However, function homogeneity between time steps is lost. This is due to the dimension of U changing to suit the input vector dimension for each time step
   3. The input variability point is also left unsatisfied, as we would not have a separate U parameter for any additional iteration.
5. Thus we have satisfied the previous input dependency point but the other two points are now not satisfied

## 

## 11.1.7: Introducing RNNs

So what is the solution?

1. The solution which satisfies all 3 conditions are **Recurrent Neural Networks (RNNs)**
   1. As we can see from the image, there is a change to the equations
      1. Here, we have added a new term ***Wsi-1***
      2. This calls for each ato be dependent on the previous values as well
      3. All the terms are all the same dimension and can hence be added.
      4. This step remains the same, except now we compute as a function of previous inputs.
2. With this modification, all three points are satisfied
   1. **Function homogeneity across time steps:** Since the parameters W, U, V, b and c remain constant, the same function is implemented for each time step.
   2. **Variation in number of inputs:** Regardless if there are 99 or 120 or 5 inputs, the network is able to handle it.
   3. **Output dependency on previous inputs:** From the new equation we can see that depends on which in turn depends on . Thus final output depends on thereby depending on .

## 

## 11.1.8: Summary and what next

Let’s look at what we have learned so far

1. We have seen a few types of sequence learning problems
   1. Character/text prediction: Predictive autocompletion of words
   2. Part of speech tagging: Identifying the grammatical tag of each word in a sentence
   3. Sentiment analysis: Where we analyse all the inputs and apply a class to the entire sequence.
2. We have found that RNNs can be used to address these sequence learning problems, while satisfying the 3 main criteria of sequence learning.
   1. **Function homogeneity across time steps**
   2. **Variation in number of inputs**
   3. **Output dependency on previous inputs**
3. What do we do next?
   1. We must learn how to represent words and characters as numbers. **(Data & Tasks)**
   2. We need to identify an appropriate loss function. **(Loss)**
   3. We need to understand how to train the model. **(Learning Algorithm)**