The sort of architecture is made for natural language processing problems where the output is a sequence of words. It is commonly used in machine translation, video captioning, and speech recognition. An illustration is already provided in Figure 36-10.

Bidirectional Recurrent Neural Networks

Bidirectional RNN is another particular type of recurrent neural network architecture that involves placing the recurrent layers beside each other where one layer works to learn the long-term dependencies from the past; this layer is called the forward LSTM. For the other layer, the input is reversed and fed into the network, so the network learns long-term dependencies from the future. This layer is called the backward LSTM. The bidirectional RNN is illustrated in Figure 36-18.

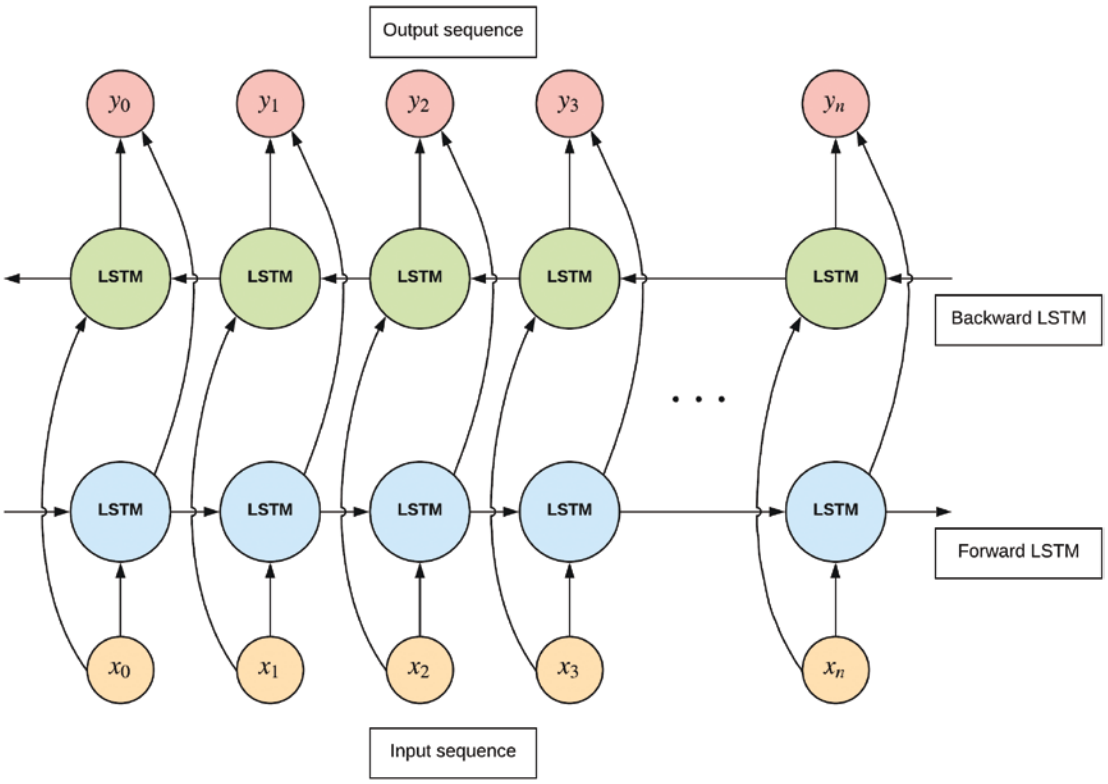

Figure 36-18. *Bidirectional LSTM*

When the outputs from these side-by-side networks are combined, it is easier to predict the next time step of a sequence having privy to the entire information gamut, because they process both information from the past and the future. Although this architecture was first designed for speech recognition tasks, it has performed impressively across a variety of other sequence prediction tasks. It is built to improve on the vanilla unidirectional LSTM which only has knowledge of the past.

This network is built on the understanding that some learning problems only make sense when a coherent set of information is present. For example, if a human interpreter is interpreting from one language to another, he first listens to a cohesive set of information in one language before interpreting to another language. This is because the context of an entire cohesive sentence gives the right basis for a correct interpretation.

RNN with TensorFlow 2.0: Univariate Timeseries

This section makes use of the Nigeria power consumption dataset to implement a univariate timeseries model with LSTM recurrent neural networks. The dataset for this example is the Nigeria power consumption data from January 1 to March 11 by Hipel and McLeod (1994), retrieved from DataMarket.

The dataset is preprocessed for timeseries modeling with RNNs by converting the data input and outputs into sequences using the method **'convert_to_ sequences'**. This method splits the dataset into rolling sequences consisting of 20 rows (or time steps) using a window of **1**}. In Figure 36-19, the example univariate dataset is converted into sequences of five time steps, where output sequence is one step ahead of the input sequence. Each sequence contains five rows (determined by the **time_steps** variable) and in this univariate case, 1 column.