

10.4 Encoder-Decoder Sequence-to-Sequence Architectures

We have seen in figure 10.5 how an RNN can map an input sequence to a fixed-size vector. We have seen in figure 10.9 how an RNN can map a fixed-size vector to a sequence. We have seen in figures 10.3, 10.4, 10.10 and 10.11 how an RNN can map an input sequence to an output sequence of the same length.

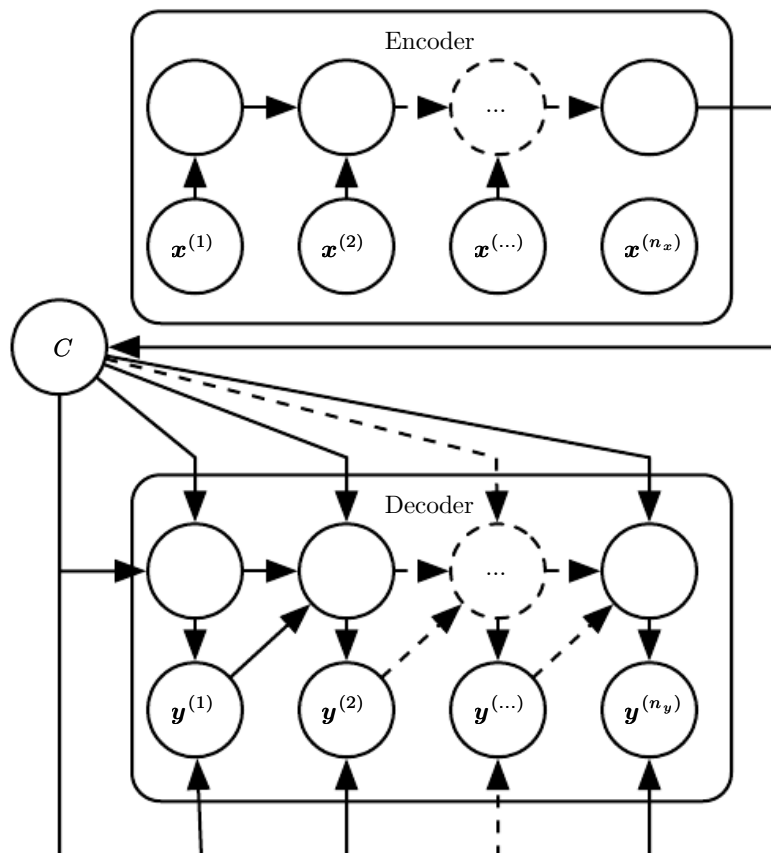


Figure 10.12: Example of an encoder-decoder or sequence-to-sequence RNN architecture, for learning to generate an output sequence $(\mathbf{y}^{(1)}, \dots, \mathbf{y}^{(n_y)})$ given an input sequence $(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(n_x)})$. It is composed of an encoder RNN that reads the input sequence and a decoder RNN that generates the output sequence (or computes the probability of a given output sequence). The final hidden state of the encoder RNN is used to compute a generally fixed-size context variable C which represents a semantic summary of the input sequence and is given as input to the decoder RNN.

Here we discuss how an RNN can be trained to map an input sequence to an output sequence which is not necessarily of the same length. This comes up in many applications, such as speech recognition, machine translation or question