sequence still has one restriction, which is that the length of both sequences must be the same. We describe how to remove this restriction in section 10.4.

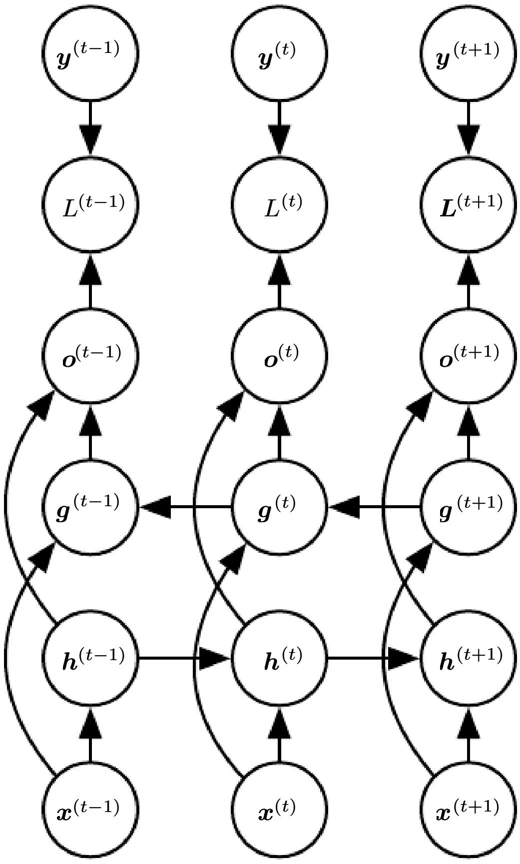

Figure 10.11: Computation of a typical bidirectional recurrent neural network, meant to learn to map input sequences $\boldsymbol{x}$ to target sequences $\boldsymbol{y}$, with loss $L^{(t)}$ at each step t. The $\boldsymbol{h}$ recurrence propagates information forward in time (towards the right) while the $\boldsymbol{g}$ recurrence propagates information backward in time (towards the left). Thus at each point t, the output units $\boldsymbol{o}^{(t)}$ can benefit from a relevant summary of the past in its $\boldsymbol{h}^{(t)}$ input and from a relevant summary of the future in its $\boldsymbol{g}^{(t)}$ input.

10.3 Bidirectional RNNs

All of the recurrent networks we have considered up to now have a "causal" structure, meaning that the state at time t only captures information from the past, $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(t-1)}$, and the present input $\boldsymbol{x}^{(t)}$. Some of the models we have discussed also allow information from past $\boldsymbol{y}$ values to affect the current state when the $\boldsymbol{y}$ values are available.

However, in many applications we want to output a prediction of $\boldsymbol{y}^{(t)}$ which may