9.7. Backpropagation Through Time

massive gradients from destabilizing training. We hinted that the exploding gradients stem from backpropagating modern RNN architectures, let's take a closer look at how *backpropagation* works in sequence models in mathematical detail. Hopefully, this discussion will bring some precision to the notion of *vanishing* and *exploding* gradients. If you recall our discussion of forward and backward propagation through computational graphs when we introduced MLPs in Section 5.3 (../chapter_multilayer-perceptrons/backprop.html#sec-backprop), then forward propagation in RNNs should be relatively straightforward. Applying backpropagation in RNNs is called *backpropagation through time* (Werbos, 1990 (../chapter_references/zreferences.html#id313)). This procedure requires us to expand (or unroll) the computational graph of an RNN one time step at a time. The unrolled RNN is essentially a feedforward neural network with the special property that the same parameters are repeated throughout the unrolled network, appearing at each time step. Then, just as in any feedforward neural network, we can apply the chain rule, backpropagating gradients through the unrolled net. The gradient with respect to each parameter must be summed across all places that the parameter occurs in the unrolled net. Handling such weight tying should be familiar from our chapters on convolutional neural networks.

Complications arise because sequences can be rather long. It is not unusual to work with text sequences consisting of over a thousand tokens. Note that this poses problems both from a computational (too much memory) and optimization (numerical instability) standpoint. Input from the first step passes through over 1000 matrix products before arriving at the output, and another 1000 matrix products are required to compute the gradient. We now analyze what can go wrong and how to address it in practice.

9.7.1. Analysis of Gradients in