

where v is the norm threshold and \mathbf{g} is used to update parameters. Because the gradient of all the parameters (including different groups of parameters, such as weights and biases) is renormalized jointly with a single scaling factor, the latter method has the advantage that it guarantees that each step is still in the gradient direction, but experiments suggest that both forms work similarly. Although the parameter update has the same direction as the true gradient, with gradient norm clipping, the parameter update vector norm is now bounded. This bounded gradient avoids performing a detrimental step when the gradient explodes. In fact, even simply taking a *random step* when the gradient magnitude is above a threshold tends to work almost as well. If the explosion is so severe that the gradient is numerically `Inf` or `Nan` (considered infinite or not-a-number), then a random step of size v can be taken and will typically move away from the numerically unstable configuration. Clipping the gradient norm per-minibatch will not change the direction of the gradient for an individual minibatch. However, taking the average of the norm-clipped gradient from many minibatches is not equivalent to clipping the norm of the true gradient (the gradient formed from using all examples). Examples that have large gradient norm, as well as examples that appear in the same minibatch as such examples, will have their contribution to the final direction diminished. This stands in contrast to traditional minibatch gradient descent, where the true gradient direction is equal to the average over all minibatch gradients. Put another way, traditional stochastic gradient descent uses an unbiased estimate of the gradient, while gradient descent with norm clipping introduces a heuristic bias that we know empirically to be useful. With element-wise clipping, the direction of the update is not aligned with the true gradient or the minibatch gradient, but it is still a descent direction. It has also been proposed (Graves, 2013) to clip the back-propagated gradient (with respect to hidden units) but no comparison has been published between these variants; we conjecture that all these methods behave similarly.

10.11.2 Regularizing to Encourage Information Flow

Gradient clipping helps to deal with exploding gradients, but it does not help with vanishing gradients. To address vanishing gradients and better capture long-term dependencies, we discussed the idea of creating paths in the computational graph of the unfolded recurrent architecture along which the product of gradients associated with arcs is near 1. One approach to achieve this is with LSTMs and other self-loops and gating mechanisms, described above in section 10.10. Another idea is to regularize or constrain the parameters so as to encourage “information flow.” In particular, we would like the gradient vector $\nabla_{\mathbf{h}(t)} L$ being back-propagated to