why the features learned by the autoencoder are useful: they describe the latent variables that explain the input.

Early work on sparse autoencoders (Ranzato *et al.*, 2007a, 2008) explored various forms of sparsity and proposed a connection between the sparsity penalty and the $\log Z$ term that arises when applying maximum likelihood to an undirected probabilistic model $p(\boldsymbol{x}) = \frac{1}{Z}\tilde{p}(\boldsymbol{x})$. The idea is that minimizing $\log Z$ prevents a probabilistic model from having high probability everywhere, and imposing sparsity on an autoencoder prevents the autoencoder from having low reconstruction error everywhere. In this case, the connection is on the level of an intuitive understanding of a general mechanism rather than a mathematical correspondence. The interpretation of the sparsity penalty as corresponding to $\log p_{\text{model}}(\boldsymbol{h})$ in a directed model $p_{\text{model}}(\boldsymbol{h})p_{\text{model}}(\boldsymbol{x} \mid \boldsymbol{h})$ is more mathematically straightforward.

One way to achieve *actual zeros* in $\boldsymbol{h}$ for sparse (and denoising) autoencoders was introduced in Glorot *et al.* (2011b). The idea is to use rectified linear units to produce the code layer. With a prior that actually pushes the representations to zero (like the absolute value penalty), one can thus indirectly control the average number of zeros in the representation.

14.2.2 Denoising Autoencoders

Rather than adding a penalty Ω to the cost function, we can obtain an autoencoder that learns something useful by changing the reconstruction error term of the cost function.

Traditionally, autoencoders minimize some function

$$L(\boldsymbol{x}, g(f(\boldsymbol{x}))) \tag{14.8}$$

where L is a loss function penalizing $g(f(\boldsymbol{x}))$ for being dissimilar from $\boldsymbol{x}$, such as the L^2 norm of their difference. This encourages $g \circ f$ to learn to be merely an identity function if they have the capacity to do so.

A **denoising autoencoder** or DAE instead minimizes

$$L(\boldsymbol{x}, g(f(\tilde{\boldsymbol{x}}))), \tag{14.9}$$

where $\tilde{\boldsymbol{x}}$ is a copy of $\boldsymbol{x}$ that has been corrupted by some form of noise. Denoising autoencoders must therefore undo this corruption rather than simply copying their input.

Denoising training forces f and g to implicitly learn the structure of $p_{\text{data}}(\boldsymbol{x})$, as shown by Alain and Bengio (2013) and Bengio *et al.* (2013c). Denoising