

the input, with no option to ignore sections of it. These tasks include the following:

- **Density estimation:** given an input \mathbf{x} , the machine learning system returns an estimate of the true density $p(\mathbf{x})$ under the data generating distribution. This requires only a single output, but it does require a complete understanding of the entire input. If even one element of the vector is unusual, the system must assign it a low probability.
- **Denoising:** given a damaged or incorrectly observed input $\tilde{\mathbf{x}}$, the machine learning system returns an estimate of the original or correct \mathbf{x} . For example, the machine learning system might be asked to remove dust or scratches from an old photograph. This requires multiple outputs (every element of the estimated clean example \mathbf{x}) and an understanding of the entire input (since even one damaged area will still reveal the final estimate as being damaged).
- **Missing value imputation:** given the observations of some elements of \mathbf{x} , the model is asked to return estimates of or a probability distribution over some or all of the unobserved elements of \mathbf{x} . This requires multiple outputs. Because the model could be asked to restore any of the elements of \mathbf{x} , it must understand the entire input.
- **Sampling:** the model generates new samples from the distribution $p(\mathbf{x})$. Applications include speech synthesis, i.e. producing new waveforms that sound like natural human speech. This requires multiple output values and a good model of the entire input. If the samples have even one element drawn from the wrong distribution, then the sampling process is wrong.

For an example of a sampling task using small natural images, see figure 16.1.

Modeling a rich distribution over thousands or millions of random variables is a challenging task, both computationally and statistically. Suppose we only wanted to model binary variables. This is the simplest possible case, and yet already it seems overwhelming. For a small, 32×32 pixel color (RGB) image, there are 2^{3072} possible binary images of this form. This number is over 10^{800} times larger than the estimated number of atoms in the universe.

In general, if we wish to model a distribution over a random vector \mathbf{x} containing n discrete variables capable of taking on k values each, then the naive approach of representing $P(\mathbf{x})$ by storing a lookup table with one probability value per possible outcome requires k^n parameters!

This is not feasible for several reasons: