

of models to those with tractable mean field fixed point equations. The variational autoencoder also has the advantage that it increases a bound on the log-likelihood of the model, while the criteria for the MP-DBM and related models are more heuristic and have little probabilistic interpretation beyond making the results of approximate inference accurate. One disadvantage of the variational autoencoder is that it learns an inference network for only one problem, inferring \mathbf{z} given \mathbf{x} . The older methods are able to perform approximate inference over any subset of variables given any other subset of variables, because the mean field fixed point equations specify how to share parameters between the computational graphs for all of these different problems.

One very nice property of the variational autoencoder is that simultaneously training a parametric encoder in combination with the generator network forces the model to learn a predictable coordinate system that the encoder can capture. This makes it an excellent manifold learning algorithm. See figure 20.6 for examples of low-dimensional manifolds learned by the variational autoencoder. In one of the cases demonstrated in the figure, the algorithm discovered two independent factors of variation present in images of faces: angle of rotation and emotional expression.

20.10.4 Generative Adversarial Networks

Generative adversarial networks or GANs (Goodfellow *et al.*, 2014c) are another generative modeling approach based on differentiable generator networks.

Generative adversarial networks are based on a game theoretic scenario in which the generator network must compete against an adversary. The generator network directly produces samples $\mathbf{x} = g(\mathbf{z}; \boldsymbol{\theta}^{(g)})$. Its adversary, the **discriminator network**, attempts to distinguish between samples drawn from the training data and samples drawn from the generator. The discriminator emits a probability value given by $d(\mathbf{x}; \boldsymbol{\theta}^{(d)})$, indicating the probability that \mathbf{x} is a real training example rather than a fake sample drawn from the model.

The simplest way to formulate learning in generative adversarial networks is as a zero-sum game, in which a function $v(\boldsymbol{\theta}^{(g)}, \boldsymbol{\theta}^{(d)})$ determines the payoff of the discriminator. The generator receives $-v(\boldsymbol{\theta}^{(g)}, \boldsymbol{\theta}^{(d)})$ as its own payoff. During learning, each player attempts to maximize its own payoff, so that at convergence

$$g^* = \arg \min_g \max_d v(g, d). \quad (20.80)$$

The default choice for v is

$$v(\boldsymbol{\theta}^{(g)}, \boldsymbol{\theta}^{(d)}) = \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}} \log d(\mathbf{x}) + \mathbb{E}_{\mathbf{x} \sim p_{\text{model}}} \log (1 - d(\mathbf{x})). \quad (20.81)$$