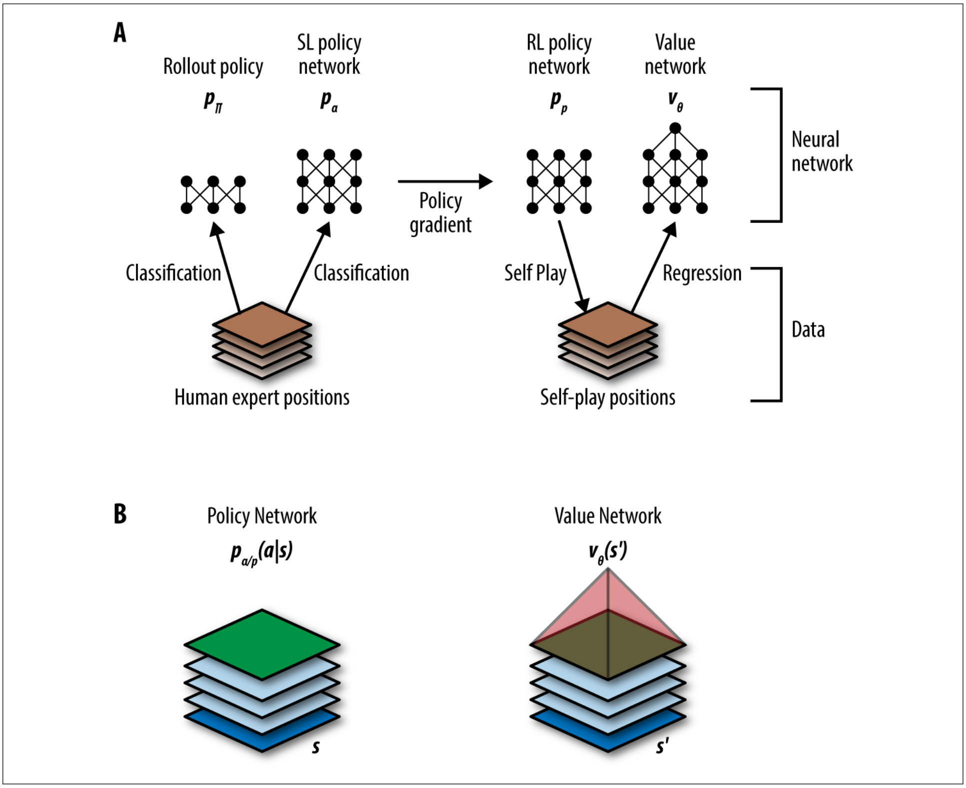

Figure 1-12. A) Depiction of AlphaGo's architecture. Initially a policy network to select moves is trained on a dataset of expert games. This policy is then refined by self-play. "RL" indicates reinforcement learning and "SL" indicates supervised learning. B) Both the policy and value networks operate on representations of the game board.

Generative Adversarial Networks

Generative adversarial networks (GANs) are a new type of deep network that uses two competing neural networks, the generator and the adversary (also called the discriminator), which duel against each other. The generator tries to draw samples from a training distribution (for example, tries to generate realistic images of birds). The discriminator works on differentiating samples drawn from the generator from true data samples. (Is a particular bird a real image or generator-created?) This "adversarial" training for GANs seems capable of generating image samples of considerably higher fidelity than other techniques and may be useful for training effective discriminators with limited data. A GAN architecture is illustrated in Figure 1-13.

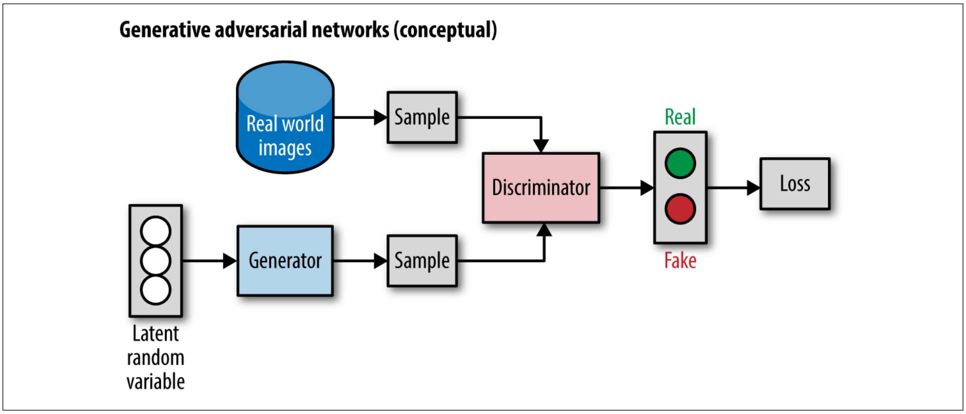

Figure 1-13. A conceptual depiction of a generative adversarial network (GAN).

GANs have proven capable of generating very realistic images, and will likely power the next generation of computer graphics tools. Samples from such systems are now approaching photorealism. However, many theoretical and practical caveats still remain to be worked out with these systems and much research is still needed.

Neural Turing Machines

Most of the deep learning systems presented so far have learned complex functions with limited domains of applicability; for example, object detection, image captioning, machine translation, or Go game-play. But, could we perhaps have deep architectures that learn general algorithmic concepts such as sorting, addition, or multiplication?

The Neural Turing machine (NTM) is a first attempt at making a deep learning architecture capable of learning arbitrary algorithms. This architecture adds an external memory bank to an LSTM-like system, to allow the deep architecture to make use of scratch space to compute more sophisticated functions. At the moment, NTM-like architectures are still quite limited, and only capable of learning simple algorithms. Nevertheless, NTM methods remain an active area of research and future advances may transform these early demonstrations into practical learning tools. The NTM architecture is conceptually illustrated in Figure 1-14.