**1. What are the main tasks that autoencoders are used for?**

* **Dimensionality reduction**: Compressing high-dimensional data into a lower-dimensional representation.
* **Denoising**: Reconstructing clean data from noisy inputs.
* **Feature extraction**: Learning useful features for downstream tasks like classification.
* **Anomaly detection**: Identifying outliers by learning the normal data distribution.
* **Data generation**: Generating new data samples (e.g., using variational autoencoders).

**2. Suppose you want to train a classifier, and you have plenty of unlabeled training data but only a few thousand labeled instances. How can autoencoders help? How would you proceed?**

* **How autoencoders help**:
  + Autoencoders can learn useful feature representations from the unlabeled data, which can then be used to improve the classifier's performance.
* **How to proceed**:
  1. Train an autoencoder on the unlabeled data to learn a compressed representation.
  2. Use the encoder part of the autoencoder to transform the labeled data into the learned feature space.
  3. Train a classifier (e.g., a simple feedforward neural network) on the transformed labeled data.

**3. If an autoencoder perfectly reconstructs the inputs, is it necessarily a good autoencoder? How can you evaluate the performance of an autoencoder?**

* **Not necessarily**:
  + Perfect reconstruction may indicate overfitting, especially if the autoencoder is overcomplete.
  + A good autoencoder should generalize well to unseen data and capture meaningful patterns.
* **Evaluation**:
  + Measure reconstruction error on a validation set.
  + For generative tasks, evaluate the quality of generated samples.
  + For feature extraction, evaluate downstream task performance (e.g., classification accuracy).

**4. What are undercomplete and overcomplete autoencoders? What is the main risk of an excessively undercomplete autoencoder? What about the main risk of an overcomplete autoencoder?**

* **Undercomplete autoencoder**:
  + The hidden layer has fewer units than the input layer.
  + **Risk**: May fail to capture important features, leading to poor reconstruction.
* **Overcomplete autoencoder**:
  + The hidden layer has more units than the input layer.
  + **Risk**: May overfit the data, learning to copy inputs without capturing meaningful patterns.

**5. How do you tie weights in a stacked autoencoder? What is the point of doing so?**

* **Tying weights**:
  + The weights of the decoder are constrained to be the transpose of the encoder's weights.
* **Purpose**:
  + Reduces the number of parameters, making the model easier to train and less prone to overfitting.
  + Encourages the autoencoder to learn a more meaningful representation.

**6. What is a generative model? Can you name a type of generative autoencoder?**

* **Generative model**:
  + A model that learns the underlying data distribution and can generate new samples from it.
* **Generative autoencoder**:
  + **Variational Autoencoder (VAE)**: A type of autoencoder that learns a probabilistic representation of the data, enabling generation of new samples.

**7. What is a GAN? Can you name a few tasks where GANs can shine?**

* **GAN (Generative Adversarial Network)**:
  + A generative model consisting of two networks: a generator (creates fake data) and a discriminator (distinguishes real from fake data).
* **Tasks**:
  + Image generation (e.g., creating realistic photos).
  + Image-to-image translation (e.g., converting sketches to photos).
  + Super-resolution (e.g., enhancing image resolution).
  + Data augmentation (e.g., generating synthetic training data).

**8. What are the main difficulties when training GANs?**

* **Mode collapse**: The generator produces limited varieties of samples.
* **Training instability**: The generator and discriminator may fail to converge.
* **Balancing**: The generator and discriminator must be balanced; otherwise, one may overpower the other.
* **Evaluation**: It is difficult to objectively evaluate the quality of generated samples.