Adversarial Autoencoders

2 Adversarial Autoencoders

- $\mathbf{x}$: input, $\mathbf{z}$: latent vector
- $p(\mathbf{z})$: prior distribution, $q(\mathbf{z}|\mathbf{x})$: encoding distribution, $p(\mathbf{x}|\mathbf{z})$: decoding distribution
- $p_d(\mathbf{x})$: data distribution, $p(\mathbf{x})$: model distribution

aggregated posterior distribution of $q(\mathbf{z})$ 은 아래와 같다.

$$q(\mathbf{z}) = \int_{\mathbf{x}} q(\mathbf{z}|\mathbf{x}) p_d(\mathbf{x}) d\mathbf{x}$$

adversarial autoencoder는 이 posterior가 prior $p(\mathbf{z})$가 되도록 regularize한다고 할 수 있다. adversarial autoencoder의 훈련과정은 아래와 같다.

- autoencoder는 reconstruction error를 낮춘다.
- adversarial network에서 generator는 autoencoder의 encoder라고 할 수 있는데 discriminator가 true라고 학습하는 prior $p(\mathbf{z})$에서 sampling하는 역할이다.
- discriminator는 generator를 통해 $p(\mathbf{z})$에서 나온 값과 autoencoder의 $q(\mathbf{z})$에서 나온 값을 분류하는 역할을 하는 것이다.

훈련이 다 끝나면 decoder는 data distribution에 prior $p(\mathbf{z})$가 가해진 generative model이 되는 것이다.

- possible choice for the encoder $q(\mathbf{z}|\mathbf{x})$
  - Deterministic
  - Gaussian posterior
  - Universal approximator posterior
- Universal approximator posterior
  - encoder network가 function $f(\mathbf{x}, \eta)$라고 하자 (input $\mathbf{x}$, random noise η with a fixed distribution)
  - $q(\mathbf{z}) = \int_{\mathbf{x}} \int_{\eta} q(\mathbf{z}|\mathbf{x}, \eta) p_d(\mathbf{x}) p_\eta(\eta) d\eta d\mathbf{x}$
  - 즉, stochasticity in $q(\mathbf{z})$는 data distribution과 random noise η에서 발생한다는 것

2.1 Relationship to Variational Autoencoders

일단 VAE의 Loss function을 다시 생각해보자.

$$L_i(\phi, \theta, \mathbf{x}_i) = -E_{\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x}_i)} \left[\log(p_\theta(\mathbf{x}_i|\mathbf{z}))\right] + KL(q_\phi(\mathbf{z}|\mathbf{x}_i) \| p(\mathbf{z}))$$

$$\text{논문에서는 } \rightarrow E_{\mathbf{x} \sim p_d(\mathbf{x})}[-\log p(\mathbf{x})] < E_{\mathbf{x}} \left[E_{q_\phi(\mathbf{z}|x)}[-\log p_\theta(\mathbf{x}|\mathbf{z})]\right] + E_{\mathbf{x}} \left[KL(q_\phi(\mathbf{z}|\mathbf{x}) \| p(\mathbf{z}))\right]$$

여기서 KL term의 특징은

- Easily draw samples from distribution

- KL divergence can be calculated

이러한 이유때문에 VAE에서 prior $p(\mathbf{z})$와 $q_\phi(\mathbf{z}|\mathbf{x})$를 Gaussian으로 놓고 풀었다. sampling하기도 쉽고 KL divergence도 closed-form으로 풀리기 때문이다. Adversarial Autoencoder는 KL divergence를 GAN의 discriminator로 바꾸어서 훈련한다. 그러면 prior에서 sampling만 가능하면 되기에 Gaussian을 써야하는 제약이 사라진다.

2.3 Incorporationg Label Information in the Adversarial Regularization

latent distribution 을 더 잘 만들기 위해 label의 정보를 추가하는 경우를 생각해보자. 논문에서는 MNIST에서 10개의 mixture를 가진 GMM model을 prior로 사용하였다. latent space에서 각 분포마다 동일한 위치는 동일한 글씨스타일(eg. 기울어진 정도 등등)을 갖는다.

- Discriminator에 prior distribution에서 뽑은 sample이 input으로 들어갈 때는 해당 sample이 어떤 label을 가져야하는지 같이 넣어준다.
- Discriminator에 posterior distribution에서 뽑은 sample이 input으로 들어갈 때는 해당 이미지에 대한 label을 같이 넣어준다.

그러면 결국 특정 label의 이미지는 latent space에서 의도된 위치(분포)로 mapping되는 것이다.

3. Likelihood Analysis of Adversarial Autoencoders

다른 model들과 likelihood를 비교하였는데 결과가 좋게 나왔다.

4. Supervised Adversarial Autoencoders

decoder에 label과 hidden code $\mathbf{z}$를 넣어서 image를 reconstruct한다. 논문 사진을 보면 각 row가 비슷한 style로 만들어졌다.

5. Semi-Supervised Adversarial Autoencoders

data가 latent class variable $\mathbf{y}$와 continuous latent variable $\mathbf{z}$로 부터 생성되었다고 하자. 각 분포는 아래와 같다.

$$p(\mathbf{y}) = \text{Cat}(\mathbf{y}) \;\; p(\mathbf{z}) = N(\mathbf{z}|\mathbf{0}, \mathbf{I})$$

semi-supervised learning을 위해 model의 architecture를 조금 바꾼다.

- encoder를 $q(\mathbf{z}, \mathbf{y}|\mathbf{x})$로 형태를 바꾼다.
- adversarial network 두 개를 사용한다.
- 하나는 $\mathbf{y}$, 다른 하나는 $\mathbf{z}$에 대한 network이다.

model training은 reconstruction, regularization, semi-supervised classification으로 나누어서 이해할 수 있다.

- reconstruction : encoder $q(\mathbf{z}, \mathbf{y}|\mathbf{x})$을 update하고 decoder를 unlabeled mini-batch의 input을 reconstruction한다.

- regularization : 두 개의 adversarial network의 discriminator를 먼저 훈련시키고 generator를 훈련시킨다.
- semi-supervised classification : autoencoder가 $q(\mathbf{y}|\mathbf{x})$를 labeled mini-batch로 cross-entropy cost 최소화하면서 update한다.

6. Unsupervised Clustering with Adversarial Autoencoders

그럼 이번에는 label없이도 좋은 representation을 학습할 수 있는지 궁금해진다. 이 때의 architecture는 semi-superviesed의 경우와 비슷하다. 달라진 점은

- semi-supervised stage를 없애서 labeled mini-batch를 학습하지 않는 것이다.
- $q(\mathbf{y}|\mathbf{x})$가 (우리가 지정한) cluster의 수를 차원으로 가지는 one-hot vectore를 예측하도록 한다.

논문에서는 MNIST를 16개의 cluster로 학습했다. 같은 숫자여도 다른 cluster에 속하면 생성된 글씨체의 style이 달랐다.

7. Dimensionality Reduction with Adversarial Autoencoders

(2008년도 t-SNE와 같은) non-parametric dimensionality reduction의 경우 새로운 data의 embedding시키는 것이 불가능하다. 여기서는 시각화를 위한 dimensionaly reduction을 소개한다.
여기서도 architecture가 약간 다르다.

- 예를 들어, m class가 있고 이를 2차원으로 나타내고 싶다.
- 나머지는 semi-supervised의 model과 동일한데
- encoder를 통해 나온 one-hot class label에 $m*2$의 matrix를 곱하고 continuous latent $\mathbf{z}$와 concat한다.