Adversarial Autoencoders

Jussi Martin

March 6, 2017

Introduction

We go through some of the main concepts presented in the paper *Adversarial Autoencoders* written by Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey.

Adversarial Autoencoders (Definitions)

Let $\mathbf{x}$ be the input vector and $\mathbf{z}$ latent code vector (hidden units) of an autoencoder with deep decoder an encoder. Let $p(\mathbf{z})$ be the distribution we want to impose on the latent vectors, $q(\mathbf{z}|\mathbf{x})$ be an encoding distribution, and $p(\mathbf{x}|\mathbf{z})$ the decoding distribution. Let also $p_{data}(\mathbf{x})$ be the data distribution, $p(\mathbf{x})$ the model distribution. The encoding function of the autoencoder $q(\mathbf{z}|\mathbf{x})$ defines an *aggregated distribution*

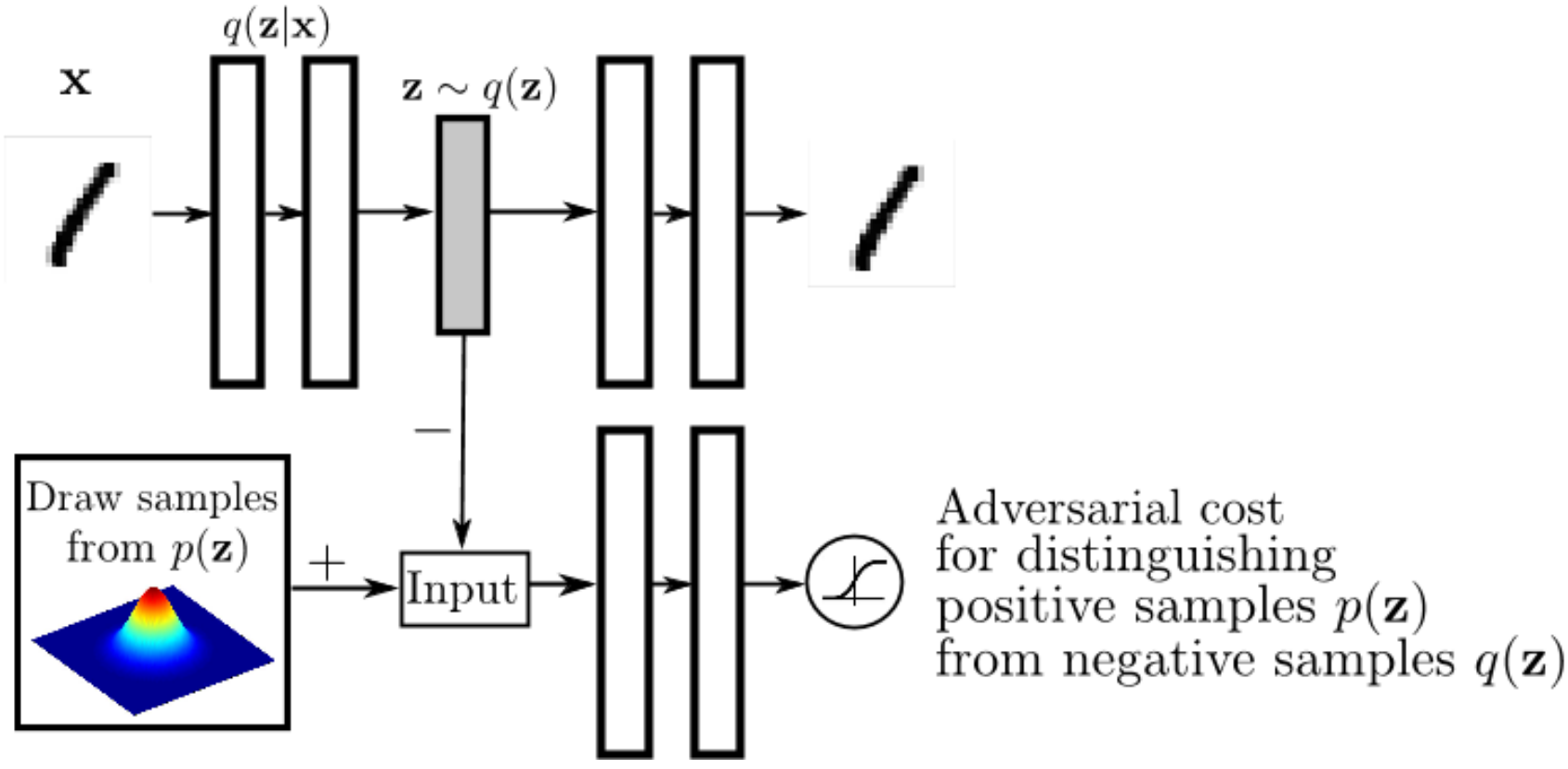
$$q(\mathbf{z}) = \int q(\mathbf{z}|\mathbf{x}) p_{data}(\mathbf{x}) d\mathbf{x}$$

on the latent vectors.

Adversarial Autoencoders (Principle)

The adversarial autoencoder is an autoencoder that is regularized by matching the aggregated posterior $q(\mathbf{z})$ to the given prior $p(\mathbf{z})$. This is done by attaching an adversarial network on top of the latent vector $\mathbf{z}$, and allowing it to guide $q(\mathbf{z})$ to converge towards $p(\mathbf{z})$. Meanwhile the autoencoder attemps to minimize the reconstruction error.
The encoder of the autoencoder is also the generator of the adversarial network and it tries to fool its discriminator in to thinking that the hidden code $q(\mathbf{z})$ comes from the true distribution $p(\mathbf{z})$.

Architecture



Training

Both the adversarial network and the autoencoder are trained jointly with SGD in two phases – the *reconstruction* phase and the *regularization* phase – executed on each mini-batch.
In the reconstruction phase, the autoencoder updates the encoder and the decoder to minimize the reconstruction error of the inputs.
In the regularization phase, the adversarial network first updates its discriminative network to tell apart true samples ($z \sim p(\mathbf{z})$) from generated samples ($z \sim q(\mathbf{z})$). Then the adversarial network updates its generator (the encoder of the autoencoder) to fool the discriminator.
When the training is finished, the decoder of the autoencoder will define a generative model that maps the imposed prior of $p(\mathbf{z})$ to the data distribution.

Different types of AAEs

Accordig to the paper the adversarial autoencoder can be:

- Deterministic
- Gaussian Posterior
- Universal Approximator Posterior

Howerver, the last approach might have some problems (see the comment in Dustin Tran's blogpost), so we present only the first two here.

Deterministic

Here we assume that $q(\mathbf{z}|\mathbf{x})$ is a deterministic function of $\mathbf{x}$. In this case the encoder is similar to the encoder of a standard autoencoder and the only source of stochasticity in $q(\mathbf{z})$ is the data distribution $p_{data}(\mathbf{x})$.

Gaussian Posterior

Here we assume that $q(\mathbf{z}|\mathbf{x})$ is a Gaussian distribution whose mean and variance is predicted by the encoder network:

$$\mathbf{z} = \begin{pmatrix} z_1 \\ \vdots \\ z_n \end{pmatrix} \sim \begin{pmatrix} \mathcal{N}(\mu_1(\mathbf{x}), \sigma_1(\mathbf{x})) \\ \vdots \\ \mathcal{N}(\mu_n(\mathbf{x}), \sigma_n(\mathbf{x})) \end{pmatrix}.$$

In this case, the stochasticity in $q(\mathbf{z})$ comes from both the data-distribution and the Gaussian distribution at the output of the encoder. We can use the same re-parametrization trick of [Kingma and Welling, 2014] for back-propagation through the encoder network.

Comparison with Variational Autoencoders

Variational autoencoders try to minimize the upper bound

$$E_{p_{data}(\mathbf{x})}[E_{q(\mathbf{z}|\mathbf{x})}[-\log p(\mathbf{x}|\mathbf{z})]] - E_{p_{data}(\mathbf{x})}[KL(q(\mathbf{z}|\mathbf{x})\|p(\mathbf{z}))]$$

of $E_{p_{data(\mathbf{x})}}[-\log p(\mathbf{x})]$. Where the first term is the reconstruction error of the decoder $p(\mathbf{x}|\mathbf{z})$ trying to evaluate codes from the encoder $q(\mathbf{z}|\mathbf{x})$. The second term is the regularizer.
The adversarial autoencoder's on the other hand uses

$$JSD(p(\mathbf{z})\|q(\mathbf{z})) = \frac{1}{2}KL\Big(p(\mathbf{z})\Big\|\frac{p(\mathbf{z})+q(\mathbf{z})}{2}\Big)+\frac{1}{2}KL\Big(q(\mathbf{z})\Big\|\frac{p(\mathbf{z})+q(\mathbf{z})}{2}\Big).$$

as a regularizer implicitly due to encoders adversarial training.

Further Topics

- Incorporating Label Information
- Semi-supervised AAEs
- Unsupervised Clustering

I might cover these in a later session.

Sources

- The paper: https://arxiv.org/pdf/1511.05644.pdf
- Dustin Tran's blogpost: http://dustintran.com/blog/adversarial-autoencoders
- Auto-Encoding Variational Bayes: https://arxiv.org/pdf/1312.6114.pdf
- Generative Adversarial Nets: https://arxiv.org/pdf/1406.2661.pdf
- Pictures from the paper: https://www.semanticscholar.org