CS231n: CNNs for Visual Recognition	**Spring 2017**
Lecture 13	
Course Coordinator: Prof. Fei-Fei Li	*Scribes: Akash Gupta*

Unsupervised learning - Unlabeled training data and the goal is to learn some underlying structure of data. E.g. - Clustering, Dimensionality Reduction, density estimation, etc.

Generative Models - Given training data, generate new samples from same distribution.

Why care? -

- Generate realistic samples, super-resolution, etc.
- Generative modeling of time-series data can be used for simulation and planning.
- Training generative models also enable inference of latent representations that can be useful as general features.

$->$ Taxonomy of Generative Models - Ian Goodfellow. Tutorial on GANs. 2017

Pixel RNN & Pixel CNN -

- Fully Visible Belief Networks.
- Explicit density model.
- Use chain rule to decompose likelihood of an image x into product of 1-d distributions and then maximize likelihood of the training data.
- Chain rule allows to define a tractable density function.

$$p(x) \;=\; \prod_{i=1}^{n} p(x_i|x_1, ..., x_{i-1})$$

Since it's a complex distribution, use neural networks to model this.

Pixel RNN -

- Generate image pixels starting from corner.
- Dependency on previous pixels modeled using an RNN(LSTM).
- sequential generation is slow.

PixelCNN -

- Still generate pixels starting from the corner.
- Dependency of previous pixels now modeled using pixels in the context region.
- Training is faster than PixelRNN. Generation time still slow since sequential generation.

Variational Autoencoders -

Background: Autoencoders - Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data. Useful for dimensionality reduction.

$$x -> z$$

z latent variable, x feature variable

$->$ Train such that latent variables can be used to reconstruct original data - "Autoencoding".

$->$ Use L2 loss for training.

$->$ Throw away the decoder and fine tune the encoder as a classifier for supervised task.

Q) *Why z should be smaller in dimension than x?*
Ans) z should represent the most important features in x OR z should capture meaningful factors of variation.

VAEs - Probabilistic spin on traditional autoencoders.

- Assume training data is generated from some underlying latent representation z.
- Representing the model -
  - Choose prior $p(z)$ to be simple. e.g. - Gaussian
  - Conditional $p(x|z)$ is complex (generates image) $->$ represent with NN - Decoder network.
- Training the model -
  - Maximize the data likelihood - (In this case prior and conditional are continuous functions)

$$p_\theta(x) = \int p_\theta(z) p_\theta(x|z) dz$$

  - Problem - Intractable function. Posterior also tractable.

$$p_\theta(z|x) = p_\theta(x|z) p_\theta(z) / p_\theta(x)$$

  - Solution - Estimate posterior using an encoder network that defines $q_\phi(z|x)$ and approximates $p_\theta(z|x)$ - Allows to derive a lower bound on data likelihood that is tractable and we can optimize.
  - Encoder Network (also called recognition/inference network) - Gives mean and covariance of $z|x$ and sample from this.
  - Decoder Network (also called generation network) - Gives mean and covariance of $x|z$ and sample from this.
- Maximizing the likelihood lower bound (also called ELBO - Evidence lower bound, since it's a value $\geq$ log of data likelihood. This is done in order to make the data likelihood a tractable function) -

$$E_z[log p_\theta(x^{(i)}|z)] - D_{KL}(q_\phi(z|x^{(i)})||p_\theta(z))$$

  - first term gives the reconstruction.
  - second term is minimizing the distance approximate dist. and prior (Gaussian in this case)
- For generation, just use the decoder on random samples drawn from prior to output new data. (Cons - Produces blurry images)

Generative Adversarial Networks (GANs) - Implicit density estimation. A game-theoretic approach.

- Problem - To generate data, sample from high dimensional training distribution. No direct wa to do this.
- Sample from a simple distribution, e.g. random noise. Learn the transformation using a NN to a training distribution.
- Represent model - 2-player game -
  - Generator network - try to fool the discriminator by generating real looking images.
  - Discriminator network - try to distinguish between real and fake images.
- Training -
  - Train jointly as minimax game.

    $$min_{\theta_g}\ max_{\theta_d}[E_{x \sim p_{data}} log D_{\theta_d}(x) + E_{z \sim p(z)} log(1 - D_{\theta_d}(G_{\theta_g}(z)))]$$

    - Discriminator (θ_d) wants to maximize objective such that $D(x)$ is close to 1(real) and $D(G(z))$ is close to 0(fake).
    - Generator (θ_g) wants to minimize objective such that $D(G(z))$ is close to 1(trying to fool the discriminator)
  - Gradient Ascent to maximize the objective for discriminator.
  - Gradient Ascent to maximize the flipped objective for generator since real objective is steep when samples are good but flat when samples are bad (small gradients). Flipping reverses this trend.
  - Algorithm - For k steps, train only the discriminator. After that update generator.
  - $k = 1$ or $k > 1$, no best rule. E.g. - Wasserstein GAN alleviates the problem, better stability.
- After training, use generator to generate new images.
- E.g. Unsupervised Representation learning with DCGAN. Radford et. al. ICLR. 2016, The GAN Zoo.
- Tips and tricks for training GANs - Link
- Beautiful, state-of-the-srt samples generated but trickier to train