April 8-12: Advanced machine learning and data analysis for the physical sciences

Morten Hjorth-Jensen[1,2]

[1]Department of Physics and Center for Computing in Science Education, University of Oslo, Norway
[2]Department of Physics and Astronomy and Facility for Rare Isotope Beams, Michigan State University, East Lansing, Michigan, USA

April 8-12, 2024

Plans for the week April 8-12

Boltzmann machines.

1. Restricted Boltzmann machines, reminder from last week
2. Variational autoencoders
3. Reading recommendation: Goodfellow et al chapters 18, 20.1-20-7

Variational Autoencoders (VAE)

Generative Models

Generative models describe a class of statistical models that are a contrast to **discriminative models**. Informally we say that generative models can generate new data instances while discriminative models discriminate between different kinds of data instances. A generative model could generate new photos of animals that look like 'real' animals while a discriminative model could tell a dog from a cat. More formally, given a data set x and a set of labels / targets y. Generative models capture the joint probability $p(x, y)$, or just $p(x)$ if there are no labels, while discriminative models capture the conditional probability $p(y|x)$. Discriminative models generally try to draw boundaries in the data space (often high dimensional), while generative models try to model how data is placed throughout the space.

Generative Adversarial Networks

Generative Adversarial Networks are a type of unsupervised machine learning algorithm proposed by Goodfellow et. al in 2014 (Read the paper first it's only 6 pages). The simplest formulation of the model is based on a game theoretic approach, *zero sum game*, where we pit two neural networks against one another. We define two rival networks, one generator g, and one discriminator d. The generator directly produces samples

$$x = g(z; \theta^{(g)}) \tag{1}$$

The discriminator attempts to distinguish between samples drawn from the training data and samples drawn from the generator. In other words, it tries to tell the difference between the fake data produced by g and the actual data samples we want to do prediction on. The discriminator outputs a probability value given by

$$d(x; \theta^{(d)}) \tag{2}$$

indicating the probability that x is a real training example rather than a fake sample the generator has generated. The simplest way to formulate the learning process in a generative adversarial network is a zero-sum game, in which a function

$$v(\theta^{(g)}, \theta^{(d)}) \tag{3}$$

determines the reward for the discriminator, while the generator gets the conjugate reward

$$-v(\theta^{(g)}, \theta^{(d)}) \tag{4}$$

During learning both of the networks maximize their own reward function, so that the generator gets better and better at tricking the discriminator, while the discriminator gets better and better at telling the difference between the fake and real data. The generator and discriminator alternate on which one trains at one time (i.e. for one epoch). In other words, we keep the generator constant and train the discriminator, then we keep the discriminator constant to train the generator and repeat. It is this back and forth dynamic which lets GANs tackle otherwise intractable generative problems. As the generator improves with training, the discriminator's performance gets worse because it cannot easily tell the difference between real and fake. If the generator ends up succeeding perfectly, the the discriminator will do no better than random guessing i.e. 50%. This progression in the training poses a problem for the convergence criteria for GANs. The discriminator feedback gets less meaningful over time, if we continue training after this point then the generator is effectively training on junk data which can undo the learning up to that point. Therefore, we stop training when the discriminator starts outputting 1/2 everywhere. At convergence we have

$$g^* = \underset{g}{\operatorname{argmin}} \max_d v(\theta^{(g)}, \theta^{(d)}) \tag{5}$$

The default choice for v is

$$v(\theta^{(g)}, \theta^{(d)}) = \mathbb{E}_{x \sim p_{\text{data}}} \log d(x) + \mathbb{E}_{x \sim p_{\text{model}}} \log(1 - d(x)) \tag{6}$$

The main motivation for the design of GANs is that the learning process requires neither approximate inference (variational autoencoders for example) nor approximation of a partition function. In the case where

$$\max_d v(\theta^{(g)}, \theta^{(d)}) \tag{7}$$

is convex in $\theta^{(g)}$ then the procedure is guaranteed to converge and is asymptotically consistent (Seth Lloyd on QuGANs). This is in general not the case and it is possible to get situations where the training process never converges because the generator and discriminator chase one another around in the parameter space indefinitely. A much deeper discussion on the currently open research problem of GAN convergence is available here. To anyone interested in learning more about GANs it is a highly recommended read. Direct quote: "In this best-performing formulation, the generator aims to increase the log probability that the discriminator makes a mistake, rather than aiming to decrease the log probability that the discriminator makes the correct prediction." Another interesting read