Notes on Deep Learning

Qi Wang

April 4, 2022

1 Basic terminologies

Closed-form Expression: A mathematical expression that consists of finite number of standard operations.(e.g. multiplications, divisions, plus, minus, etc.)
Manifold: It can be referred to a connected region. The concept of neighborhood point implies the possible existing tranformations to move from point to point. It can also be though as a small number of degree of freedom embedded in high dimensionalities of data, where each dimensionality corresponds to local direction of variantion. **Attention:**

1. *def.1* attention: using techniques to tell the algorithm what's important in a input.
2. *def.2* post-hoc attention:

Convexity:
Tractability:
Consistency: The bias between estimated result and its real value, which can be diminished as data example grows.
generalization: The ability to perform well on previously unseen data is called generalization. That is, test error reflects generalization ability.
underfitting: When the model is not able to obtain a sufficient low training error;
overfitting: When the difference between *test error* and *training error* is too large;
Regularization: This is any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error;

2 Basic machine learning

Point estimator: The attempt to provide the single best prediction of some quantity of interest, weights of model for example. Denoted as:

$$\hat{\theta_m} = g(x^{(1)}, \ldots, x^{(m)})$$

where $\hat{\theta_m}$ represents the estimated value of the interest.(e.g. weights of model)
Bias: It is defined as:

$$bias(\hat{\theta_m}) = \mathbb{E}(\hat{\theta_m}) - \theta_m$$

where θ_m is the underlying true parameter of interest.
Variance:
Maximum Likelihood Estimation: Let $p_{model}(x;\theta)$ be a parametric probability distribution, mapping any configuration of x to a real number estimating the true probability $p_{data}(x)$. That is defined as:

$$\begin{aligned}\theta_{ML} &= \operatorname*{argmax}_{\theta} p_{model}(\mathbb{X};\theta), \\ &= \operatorname*{argmax}_{\theta} \prod_{i=1}^{m} p_{model}(x^{(i)};\theta)\end{aligned}$$

Expressed in *log* scale for the simplicity of calculation:

$$\theta_{ML} = \operatorname*{argmax}_{\theta} \sum_{i=1}^{m} \log p_{model}(x^{(i)};\theta)$$

Because argmax does not change when rescaling the cost function, we can divide by m to obtain expectation version of criterion w.r.t. empirical distribution $\hat{p}_{data}$:

$$\theta_{ML} = \operatorname*{argmax}_{\theta} \mathbb{E}_{x\sim\hat{p}_{data}} \log p_{model}(x;\theta)$$

Maximum Likelihood Estimation can be viewed as minimizing the KL divergence here:

$$D_{KL}(\hat{p}_{data}||p_{model}) = \mathbb{E}_{x\sim\hat{p}_{data}(x)}[\log \hat{p}_{data}(x) - \log p_{model}(x)]$$

Since $\log \hat{p}_{data}$ is only about data generation, thus only need to minimize $\log p_{model}(x)$

Properties of Maximum Likelihood

As the number of examples $m \to \infty$, the maximum likelihood of estimate approaches the real value of the parameter. These conditions applies:

1. The true distribution p_{data} must lie within the model families $p_{model}(x;\theta)$, otherwise no estimator could recover p_{data}

2. The true distribution must correspond to only one value of parameter θ, otherwise the estimator can recover the correct p_{data} but not determining which θ

Bayesian statistics

Other than estimating single θ, another approach is to consider all possible θ values when making prediction over observed data, this is the how *Bayesian statistic* work.

One could recover the belief about θ over observed samples by applying Baysian rule:

$$p(\theta|x^{(1)},\ldots,x^{(m)}) = \frac{p(x^{(1)},\ldots,x^{(m)}|\theta)p(\theta)}{p(x^{(1)},\ldots,x^{(m)})} \tag{1}$$

One could estimate the next $m+1$ sample from observed m samples, based on our belief about θ :

$$p(x^{(m+1)}|x^{(1)},\ldots,x^{(m)}) = \int p(x^{(m+1)}|\theta)p(\theta|x^{(1)},\ldots,x^{(m)})d\theta \tag{2}$$

Thus the prediction on the next data $x^{(m+1)}$ is predicted and weighted on the posterior distribution of θ(the posterior: $p(\theta|x^{(1)},\ldots,x^{(m)})$).

Unlike the **MLL** to predict one point of the parameter, Bayesian stat. recover dataset by intergral over possible parameters. Secondly, Bayesian stat.depends on predefined prior of θ, which was initiated by large entropy and decreased to smaller and more prone to data distribution entropy as data grows.

Overall, it produce much better result given limited number of data, but requires more computation cost.

3 Feedforward neural network

This category usually refers to networks other than recurrent neural networks, which use network's output as other layer's input. The most common type of Feedforward network is MLP(multilayer perceptron) and CNN.

3.1 Weights and Bias

The gradient descent method is quite sensitive to initialization of weights, thus one shall always choose small random values for weights initialization; bias can be initialized to zero of small positive value.

Weights

In MLP, the weight refer to the weight factor in each connected neuron, e.g. from N-neuron first layer to M-neuron second layer, $N \times M$ weights were created; while due to slightly different definition of neuron in Convolutional networks, which defines each Conv filter as a neuron, weight equals to the tensor of Conv filter(e.g. a Conv2d(kernel=(2,2),input_channel=1,output_channel16), where 16 neurons each one will contain $2 \times 2 \times 1$ weights and one bias)

4 GAN

GAN compares to the rest of the generative model, differing by a two-player game strategy. In the game, the discriminator tries to distinguish generated results from real data distribution, thus to encourage the generator to produce samples closer to the real data.

Discriminator derivation

Discriminator is often regarded as "classification model" that assign image to labels, here for example "fake" and "real" labels. By means of MLE, a discriminator tries to maximize its log-likelihood of:

$$\underset{D(x)}{\operatorname{argmax}} \mathbb{E}_{x \sim p_r(x)} \log D(x) \tag{3}$$

While maximzing accuracy for identifying data from real data, discriminator maximizes accuracy of determining generated samples are false(reversely judge probability of generated samples):

$$\underset{D(x)}{\operatorname{argmax}} \mathbb{E}_{z \sim p_z(z)} \log \left(1 - D(G(z))\right) \tag{4}$$

Generator derivative

Simaltanously, for the generator to update, the goal is to decrease the accuracy of generated samples predicted by discriminator:

$$\underset{G(z)}{\operatorname{argmin}} \mathbb{E}_{z \sim p_z(z)} \log\left(1 - D(G(z))\right) \tag{5}$$

Thus a overall cost function is given to train both *discriminator* and *generator*, and this is where the both network reaches their optimal, i.e. *Nash Equilibrium*:

$$L(G, D) = \underset{G(x)}{\operatorname{argmin}} \underset{D(x)}{\operatorname{argmax}} \mathbb{E}_{x \sim p_r(x)} \log D(x) + \mathbb{E}_{x \sim p_g(x)} \log\left(1 - D(x)\right) \tag{6}$$

where $z \sim p_z(z)$ is replaced by $x \sim p_g(x)$, indicating x sampled from generated data, and $G(x)$ tries to minimize the function at RHS, while the $D(x)$ tries to maximize it.

Discriminator optimal

By calculating partial derivative of D, we know that optimal value of discriminator(D^*) is:

$$\begin{aligned} D^* &= \frac{p_r(z)}{p_r(x) + p_g(x)} \\ &= \frac{1}{2}, \end{aligned} \tag{7}$$

when $p_r(x) = p_g(x), x \in [0, 1]$.

Global optimal

Plugging in optimal value of the *discriminator*(D^*) into eq.4:

$$L(G, D^*) = \int_x p_r(x) \log D^*(x) + p_g(x) \log\left(1 - D(x)\right) dx \tag{8}$$

$$= \log(\frac{1}{2}) \int_x p_r(x) dx + \log(\frac{1}{2}) \int_x p_g(x) dx \tag{9}$$

$$= -2 \log 2 \tag{10}$$

Thus, the global optimal is obtained after optimal of *discriminator*.

Loss function

GAN was designed origionally using JS divergence, between p_r and p_g:

$$D_{JS}(p_r||p_g) = \frac{1}{2}D_{KL}(p_r||\frac{p_r+p_g}{2}) + \frac{1}{2}D_{KL}(p_g||\frac{p_r+p_g}{2}) \tag{11}$$

$$= \log 2 + \frac{1}{2}(\int_x p_r(x)\log\frac{p_r(x)}{p_r(x)+p_g(x)} \tag{12}$$

$$+ \int_x p_g(x)\log\frac{p_g(x)}{p_r(x)+p_g(x)}) \tag{13}$$

$$= \frac{1}{2}(\log 4 + L(G, D^*)) \tag{14}$$

Meaning that the most optimal *generator* replicates the real data distribution, and resulting global minimum of $L(G^*, D^*) = -2\log(2)$ inline with eq.8.

Lipschitz continuity

This is used to identity the continuity and but not necessarily differentiable of a function(e.g. $f(x) = |x|$). Defined as:

$$\exists K, \forall x_i, |f(x_1) - f(x_2)| \leq K|x_1 - x_2|$$

5 Markov Chain Monte Carlo

5.1 Markov Chain

This method applys to the following rule:

$$P_r(x_{t+1}|x_1, x_2, \ldots, x_t) = P_r(x_{t+1}|x_t)$$

Thus, the probability of next state to happen depends on current state, even though lots of states were taken places.

5.2 Monte Carlo Simulation

Monte Carlo simulation uses randomness to repeatedly calculate numerical results, restricted by a deterministic computation. Commonly, the Monte Carlo method could be applied to tackle problem that is not analytical tractable. This method is widely applied in a broad varity and have a particular pattern:

1. Define a domain of possible input;
2. Generate inputs from probability distribution randomly over the domain;
3. Perform a deterministic computation on the inputs;
4. Aggregate the inputs

6 VAE

6.1 Latent variable models

Suppose one wish to construct a image($x, x \in \mathbb{R}^D$), but do not know the distribution of it($p(x)$). We could induce another latent variable($z, z \in \mathbb{R}^d$) to account for it:

$$p(x) = \int p(x|z)p(z)dz \tag{15}$$

And find out the optimal distribution of $p_\theta(x)$:

$$\begin{aligned}
\theta^* &= \operatorname{argmax}_\theta p_\theta(x) && (16)\\
&= \mathbb{E}_{p(x)}[p_\theta(x|z)] && (17)\\
&= \mathbb{E}_{p(x)}[\frac{q_\theta(z|x)}{q_\theta(z|x)}p_\theta(x|z)] && (18)\\
&= \mathbb{E}_{q_\phi(z|x)}[\frac{p_\theta(x|z)p(z)}{q_\phi(z|x)}] && (19)
\end{aligned}$$

Eq.17 applies the *importance sampling* strategy, which approximates target distribution by weighting different inferencing distribution.

6.2 VAE components

1. $p(x, z)$ which is the generative model, consists of:
    * $p_\theta(x|z)$ a probability decoder
    * $p(z)$ a prior over latent variables
2. $q_\phi(z|x)$ a probability encoder

To approximates the posterior, one can use *KL Divergence* :

$$\begin{aligned}
D_{KL}(q_\phi(z|x)||p_\theta(z|x)) &= \mathbb{E}_{q_\phi(z|x)}[\log \frac{q_\phi(z|x)}{p_\theta(z|x)}] && (20)\\
&= \mathbb{E}_{q_\phi(z|x)}[\log q_\phi(z|x) - \log p_\theta(z, x)] + \log p_\theta(x) && (21)\\
&= -\mathcal{L}(x; \theta, \phi) + \log p_\theta(x) && (22)
\end{aligned}$$

Eq. 19 applyies *Bayesian rule* and the fact that x and z are i.i.d. sampled. The $\mathcal{L}$ is so-called **ELBO**(evidence-lower bound). Thus, as **ELBO** increases, the *log-likelihood* of $p_\theta(x)$ also increases.