Lecture 1

§1 Deep learning 的介绍

1. Deep learning 的类别

① deep supervised learning

② deep unsupervised learning

③ deep reinforcement learning

注：其中 deep supervised learning 更适合数学分析，也是本课重点

2. Deep learning 的基本设置

① input data 的维度：$d \in N = \{1, 2, 3, \cdots\}$

② available input-output data pair 的数量：$M \in N = \{1, 2, 3, \cdots\}$

③ 联系 input 和 output 的 unknown function：$\mathcal{E} \in C(R^d, R)$，

满足 $y_m = \mathcal{E}(x_m)$，$\forall\, m \in \{1, 2, \cdots, M\}$

④ avaiable input data：$x_1, x_2, \cdots, x_{M+1} \in R^d$

⑤ available output data：$y_1, y_2, \cdots, y_M \in R$

注：$C(R^d, R)$ 表示 $R^d \to R$ 的 continuous function 集合

3. Deep learning 的目标

利用前 M 组 available input-output data pair 的信息：

$$(x_1, y_1) = (x_1, \mathcal{E}(x_1)),\ (x_2, y_2) = (x_2, \mathcal{E}(x_2)),\ \cdots,\ (x_M, y_M) = (x_M, \mathcal{E}(x_M)) \in R^d \times R$$

来近似计算第 $(M+1)$ 个 input data x_{M+1} 对应的 output y_{M+1}

4. 问题的等价转换

问题等价于：求出 $\mathcal{L}: C(R^d, R) \to [0, \infty)$ 的 minimizer ϕ^*，其中

$$\mathcal{L}(\phi) = \frac{1}{M}\left[\sum_{m=1}^{M} |\phi(x_m) - y_m|^2\right],\quad \forall\, \phi \in C(R^d, R)$$ (mean square error function)

其中 $\mathcal{L}(\mathcal{E}) = 0$，因此 $\mathcal{E}: R^d \to R$ 即为 minimizer

注：由于 $\mathcal{L}$ 被定义在无穷维向量空间 $C(R^d, R)$ 上，计算机的 discrete numerical operation 不适用于该优化问题

5. 问题的近似

为了解决上述问题，将优化对象转变为有限多个参数 $\theta \in R^\partial$：

令 ① $\partial \in N$，（参数的个数）

② $\psi = (\psi_\theta)_{\theta \in R^\partial}: R^\partial \to C(R^d, R)$

③ $\mathcal{L}: R^\partial \to [0, \infty)$ 满足：$\mathcal{L} = \mathcal{L} \circ \psi = \mathcal{L}(\psi(\theta))$

注：① ψ 将 R^∂ 中的所有 θ map 到 $C(R^d, R)$ 中的部分/全部函数

② ψ 可理解为神经网络除 loss function 外的部分，用一堆参数 θ 来近似一个 $C(R^d, R)$ 中的函数

③ $\mathcal{L}$ 表示先把 θ map 成一个函数 $\psi(\theta)$，再用 $\mathcal{L}$ 来衡量 $\psi(\theta)$ 的 loss

则集合 $\{\psi_\theta : \theta \in \mathbb{R}^\vartheta\} \subseteq C(\mathbb{R}^d, \mathbb{R})$ 被用于近似无穷维向量空间 $C(\mathbb{R}^d, \mathbb{R})$

eg. $d=1$, $\vartheta=3$, 此时 $x \in \mathbb{R}^1$, $\theta=(\theta_1, \theta_2, \theta_3) \in \mathbb{R}^3$,

一个可能的 $\psi_\theta(x)$ 为 $\psi_\theta(x) = \theta_1 + \theta_2 x + \theta_3 x^2$, 但在 deep supervised learning 中, $\psi_\theta(x)$ 的形式与 ANN 有关

Taking the set in (6) and its parametrization function in (7) into account, we then intend to compute approximate minimizers of the function $\mathfrak{L}$ restricted to the set $\{\psi_\theta : \theta \in \mathbb{R}^\vartheta\}$, that is, we consider the optimization problem of computing approximate minimizers of the function

$$\{\psi_\theta : \theta \in \mathbb{R}^\vartheta\} \ni \phi \mapsto \mathfrak{L}(\phi) = \frac{1}{M}\left[\sum_{m=1}^{M} |\phi(x_m) - y_m|^2\right] \in [0, \infty). \tag{9}$$

Employing the parametrization function in (7), one can also reformulate the optimization problem in (9) as the optimization problem of computing approximate minimizers of the function

$$\mathbb{R}^\vartheta \ni \theta \mapsto \mathscr{L}(\theta) = \mathfrak{L}(\psi_\theta) = \frac{1}{M}\left[\sum_{m=1}^{M} |\psi_\theta(x_m) - y_m|^2\right] \in [0, \infty) \tag{10}$$

and this optimization problem now has the potential to be amenable for discrete numerical computations. In the context of deep supervised learning, where one chooses the parametrization function in (7) as deep ANN parametrizations, one would apply an *stochastic gradient descent* (SGD)-type optimization algorithm to the optimization problem in (10) to compute approximate minimizers of (10). In ?? in Part III we present the most common variants of such SGD-type optimization algorithms. If $\vartheta \in \mathbb{R}^\vartheta$ is an approximate minimizer of (10) in the sense that $\mathscr{L}(\vartheta) \approx \inf_{\theta \in \mathbb{R}^\vartheta} \mathscr{L}(\theta)$, one then considers $\psi_\vartheta(x_{M+1})$ as an approximation

$$\psi_\vartheta(x_{M+1}) \approx \mathcal{E}(x_{M+1}) \tag{11}$$

of the unknown output $\mathcal{E}(x_{M+1})$ of the $(M+1)$-th input data x_{M+1}. We note that in deep supervised learning algorithms one typically aims to compute an approximate minimizer $\vartheta \in \mathbb{R}^\vartheta$ of (10) in the sense that $\mathscr{L}(\vartheta) \approx \inf_{\theta \in \mathbb{R}^\vartheta} \mathscr{L}(\theta)$, which is, however, typically not a minimizer of (10) in the sense that $\mathscr{L}(\vartheta) = \inf_{\theta \in \mathbb{R}^\vartheta} \mathscr{L}(\theta)$ (cf. ??).

In (3) above we have set up an optimization problem for the learning problem by using the standard mean squared error function to measure the loss. This *mean squared error loss function* is just one possible example in the formulation of deep learning optimization problems. In particular, in image classification problems other loss functions such as the *cross-entropy loss function* are often used and we refer to Chapter 5 of Part III for a survey of commonly used loss function in deep learning algorithms (see Section 5.4.2). We also refer to ?? for convergence results in the above framework where the parametrization function in (7) corresponds to *fully-connected feedforward* ANNs (see ??).