AI1811-Deep Learning

Xiyuan Yang
2025.11.20

Lecture notes for deep learning AI-1811.

目录

§1. Introduction

§1.1. Lecture Overview

- Introduction
  - High dimensional data
  - 高维数据中的维数灾难
- Model Architecture
  - MLP
  - CNN
  - RNN
  - Transformer
- Advanced about deep learning
  - LLM reasoning
  - 凝聚现象

§1.2. NN and Polynomial

深度学习有万有逼近定理，多项式拟合也有 Weierstrass 逼近定理。

> **Recordings** The bitter lesson.
> The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin.

在实际情况下，我们发现神经网络的过拟合现象远小于多项式拟合。

§2. High Dimensional Space

§2.1. 高维数据的空间特点

$$x = [x_1, x_2, ..., x_d]^T \in \mathbb{R}^d$$

§2.1.1. 高维空间的稀疏性

Input n points in the cube in d dimensional space, with k sub-length: k^d

If we ensure that all the little cube has at least on data point, then wen must ensure:

$$n \geq k^d$$

The increase of n is exponential!

> **Recordings** High dimensional data is very sparse.
> For the sampling process in high dimensional space, the state space si extremely large.

§2.1.2. 体积集中在表面

Ω with high dimensional space for dimensional d, scale the Ω with factor ε for a little bit. We can get the inner part $(1-\varepsilon)\Omega$.

$$\frac{V(1-\varepsilon)\Omega}{V(\Omega)} = (1-\varepsilon)^d \to 0$$

This will lead to the lack of points of inner parts while sampling, which will cause the generalization error for the inner part is extremely big.

§2.1.3. 距离的集中效应

Randomly sample 2 spaces for point x and y, then the distance can be written into:

$$\|x-y\|_2 = \left(\sum_{i=1}^{d}(x_i - y_i)^2\right)^{\frac{1}{2}}$$

While d becomes extremely large, the distance will converge into $\sqrt{2}R$.

- Angle will converge into 90.
- Distance will converge into R (in the interface).

> **Recordings** Space in a ball.
> - 体积集中的表面

- 体积集中在赤道

§2.1.3.1. Gaussian Distribution in High Dimensional Space

For gaussian distribution $x \sim N(0, I_d)$

$$p(x) = \frac{1}{(2\pi)^{\frac{d}{2}}} \exp\left(-\frac{\|x\|^2}{2}\right)$$

Recordings 概率和概率密度.
- 高斯分布可以保证概率密度的分布仅是呈现正态分布的形式。
- 但是具体的概率上概率密度的积分，而在高维空间下体积分布上不均匀的。

$$F(r) = \exp\left(-\frac{r^2}{2}\right) r^{d-1}$$

The maximum point is in $r = \sqrt{d-1}$.

§2.1.3.2. Word Embedding

Word Embedding for NLP: dimension reduction in high dimensional space.

§2.1.4. 高维数据的线性可分性

线性可分性：空间中存在超平面，将不同类别的数据点完全分开。

- In kernelled functions, we will do a none-linear transformation for original low dimensional space data and embed them into high dimensional space. Then linear classifiers like SVM can be applied!
- In NN, the last layer will make the high dimensional data can be linear classified, which can be output and categorized for softmax.

§2.2. 维数灾难

Curse of dimensionality.

For example, the math expression for 2-layer neural network:

$$f(x;\theta) = \sum_{k=1}^{m} a_k \sigma(w_k^T x) = a^T \sigma(Wx)$$

§3. Recurrent Neural Network

循环神经网络相信状态随着时间的变化，因此训练可学习矩阵来表示隐状态：

$$S_0 = 0$$
$$S_t = f(Ux_t + WS_{t-1} + b), t = 1, 2, 3, ..., n$$
$$o_t = g(Vs_t)$$

where f is the activate functions, and g is the softmax functions for final output.

For each state, S_t represents the memory statement for current state t.

From this statement, we can design **encoder-decoder** structure for machine translation.

§3.1. BackPropagation Through Time (BPTT)

$$\frac{\partial L_t}{\partial W_{hh}} = \sum_{k=1}^{t} \frac{\partial L_t}{\partial h_t} \cdot \left(\prod_{j=k}^{t-1} \frac{\partial h_{j+1}}{\partial h_j} \right) \cdot \frac{\partial h_k}{\partial W_{hh}}$$

While doing back propagations, the computation of gradient requires many time steps multiplications, which will result to **Vanishing Gradient** (VG) or **Exploding Gradient** (EG).

§3.2. LSTM

We have a sequence: $x_1, x_2, ..., x_n$.

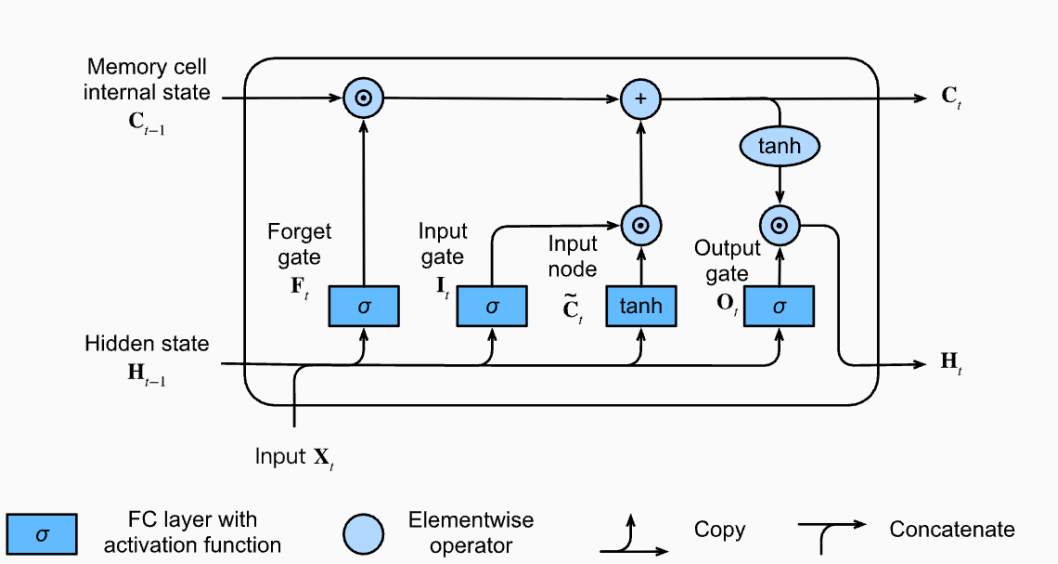

Fig. 10.1.4 Computing the hidden state in an LSTM model.

For current state t, we will do operations as follows:

- Input new knowledge:

$$\tilde{C}_{t+1} = \tanh(W_{sc} S_t + W_{xC} x_{t+1} + b_C)$$

- Choose to forget:

$$F_t = \sigma(W_{Sf} S_t + W_{xf} x_{t+1} b_f)$$

- Choose to write into long-term memory:

$$I_t = \sigma(W_{Si} S_t + W_{xi} x_{t+1} + b_i)$$

Then, we can update our memory status:

$$C_{t+1} = I_t \tilde{C}_{t+1} + F_t C_t$$

Now, we consider the output gate:

$$s_{t+1} = o_t * \tanh(C_{t+1})$$

$$o_t = \sigma(W_{so}S_t + W_{xo}x_{t+1} + b_o)$$

Recordings Why two state?.

Why we design the internal memory state and hidden state?

- 记忆门的控制代表着模型的输入信息如何更新到长期的记忆中，我们可以从公式中看到，记忆门的状态矩阵的更新是不需要通过激活函数的
- 隐藏状态在 LSTM 中代表浅层记忆，控制输入和输出，并且每一步都会重新计算，快速改变

$f_t \in [0, 1]$. Thus it could control the gradient in a relative bounds.

$$\frac{\partial L}{\partial C_t} = \frac{\partial L}{\partial C_{t+1}} * \frac{\partial C_{t+1}}{\partial C_t}$$

$$\frac{\partial C_{t+1}}{\partial C_t} = f_t \in [0, 1]$$

Thus, we can reduce VG and EG for we can controlling the gradient compute into a relative medium range.