

# 人工智能中的编程 hw2 实验报告

姓名: 杨晗  
学号: 2300011177

## 核心函数设计

### 全连接层

`forward_fc` 直接以 `cublasSgemm` 计算 (`in_features` × `batch`) 输入与权重乘积, 再用一次额外 GEMM 将 `bias` broadcast; `backward_fc` 复用三次 `cublasSgemm/Sgemv` 依次求得  $\partial X$ 、 $\partial W$  与  $\partial b$ , 完全依赖 BLAS 原语, 避免线程级实现。

### 卷积层

`im2col_kernel` 将输入 `patch` 展平后供 `conv2d_forward` 调用 `cublasSgemm` 完成  $W * X_{col}$ , 并使用一个 `ones` GEMM 添加 `bias`; `conv2d_backward` 先以 GEMM 求  $\partial W$ 、 $\partial b$ , 再把  $\partial X_{col}$  通过 `col2im_kernel` 聚合回输入, 最后一次 GEMM 特意对 `weights` 使用 `CUBLAS_OP_T`, 保证输入梯度对应  $W^T$ 。

### 池化层

`max_pool_forward_kernel` 在 `CUDA_KERNEL_LOOP` 里遍历输出, 每个线程扫描  $2 \times 2$  window 寻找最大值并记录 `argmax mask`; `max_pool_backward_kernel` 根据 `mask` 用 `atomicAdd` 将上游梯度散射回输入, 外围函数负责 `kernel launch` 与反传前的 `cudaMemset`。

### Softmax 层

Softmax 由 `row_max_kernel`、`row_sum_kernel` 两个归约 kernel 配合 `subtract_max_kernel`、`exp_kernel`、`normalize_kernel` 完成减最大值、指数和归一化, 整个过程保持在 GPU 内以保证数值稳定。

### Cross Entropy Loss

`cross_entropy_loss_kernel` 直接对 softmax 概率按标签取负对数并求平均, `softmax_cross_entropy_backward_kernel` 则计算 `probs - one_hot(labels)` 并按 `batch` 缩放, 给出与 softmax 输出同形状的梯度。

## 正确性检测

自定义一些简单数据如下, 手动计算出理论值, 用来检测核心函数正确性。

### 全连接层

1	<code>X = [[1, 2, 3],</code>
---	------------------------------

```

2      [4, 5, 6]]
3  W = [[1, 0, 1],
4        [0, 1, 1],
5        [1, 1, 0],
6        [0, 0, 1]]
7  b = [0.5, 0.5, 0.5, 0.5]
8  Y_expected = [4.5, 5.5, 3.5, 3.5,
9                10.5, 11.5, 9.5, 6.5]
10 grad_out = [0.1, 0.2, 0.3, 0.4,
11             0.5, 0.6, 0.7, 0.8]
12 dX_expected = [0.4, 0.5, 0.70000005,
13                1.2, 1.3, 1.90000001]
14 dW_expected = [[2.1, 2.7, 3.3],
15                [2.6, 3.4, 4.2],
16                [3.1, 4.1, 5.1],
17                [3.6, 4.8, 6.0]]
18 db_expected = [0.6, 0.8, 1.0, 1.2]
19 → `Test1/2 passed!`

```

## 卷积层

```

1  input = [[ 1,  2,  3,  4],
2            [ 5,  6,  7,  8],
3            [ 9, 10, 11, 12],
4            [13, 14, 15, 16]]
5  kernel = [[ 1, 0, -1],
6            [ 1, 0, -1],
7            [ 1, 0, -1]]
8  bias = 0.5
9  output_expected = [-7.5, -3.5, -3.5, 10.5,
10                    -17.5, -5.5, -5.5, 21.5,
11                    -29.5, -5.5, -5.5, 33.5,
12                    -23.5, -3.5, -3.5, 26.5]
13 grad_output = [0.1, 0.2, 0.3, 0.4,
14                0.5, 0.6, 0.7, 0.8,
15                0.9, 1.0, 1.1, 1.2,
16                1.3, 1.4, 1.5, 1.6]
17 dX_expected = [0.8, 0.4, 0.4, -1.0,
18                1.8, 0.6, 0.6, -2.1,
19                3.0, 0.600000014, 0.6, -3.3,
20                2.4, 0.40000001, 0.40000001, -2.6]
21 dW_expected = [69.6, 96.2, 73.2,
22                111.2, 149.6, 111.2000005,
23                73.2, 96.2, 69.6]
24 db_expected = 13.6
25 → `Conv im2col test passed!`

```

## 池化层

```

1 input = [[ 1, 2, 3, 4],
2           [ 5, 6, 7, 8],
3           [ 9, 10, 11, 12],
4           [13, 14, 15, 16]]
5 kernel = 2, stride = 2
6 output_expected = [6, 8, 14, 16]
7 grad_out = [1, 2, 3, 4]
8 grad_in_expected = [[0, 0, 0, 0],
9                      [0, 1, 0, 2],
10                     [0, 0, 0, 0],
11                     [0, 3, 0, 4]]
12 → `MaxPool forward/backward test passed!`

```

## Softmax 层

```

1 logits = [[1, 2, 3],
2            [1, 2, 4]]
3 softmax_expected = [0.09003058, 0.24472848, 0.66524094,
4                     0.04201007, 0.1141952, 0.8437947]
5 → `Softmax forward test passed!`

```

## Cross Entropy Loss

```

1 logits = [[1, 2, 3],
2            [1, 2, 4]]
3 labels = [2, 1]
4 loss_expected = 1.288726
5 grad_expected = [0.04501529, 0.12236424, -0.16737953,
6                  0.02100503, -0.4429024, 0.42189735]
7 → `Softmax + CrossEntropy test passed!`

```

## 正确性检测结果图

```

(base) PS D:\desktop\code\Programming-in-Artificial-Intelligence> ./hw3/build/main.exe
Test1 passed!
Test2 passed!
MaxPool forward test passed!
MaxPool backward test passed!
Softmax + CrossEntropy test passed!
Conv im2col test passed!
Softmax forward test passed!
Softmax + CrossEntropy test passed!

```