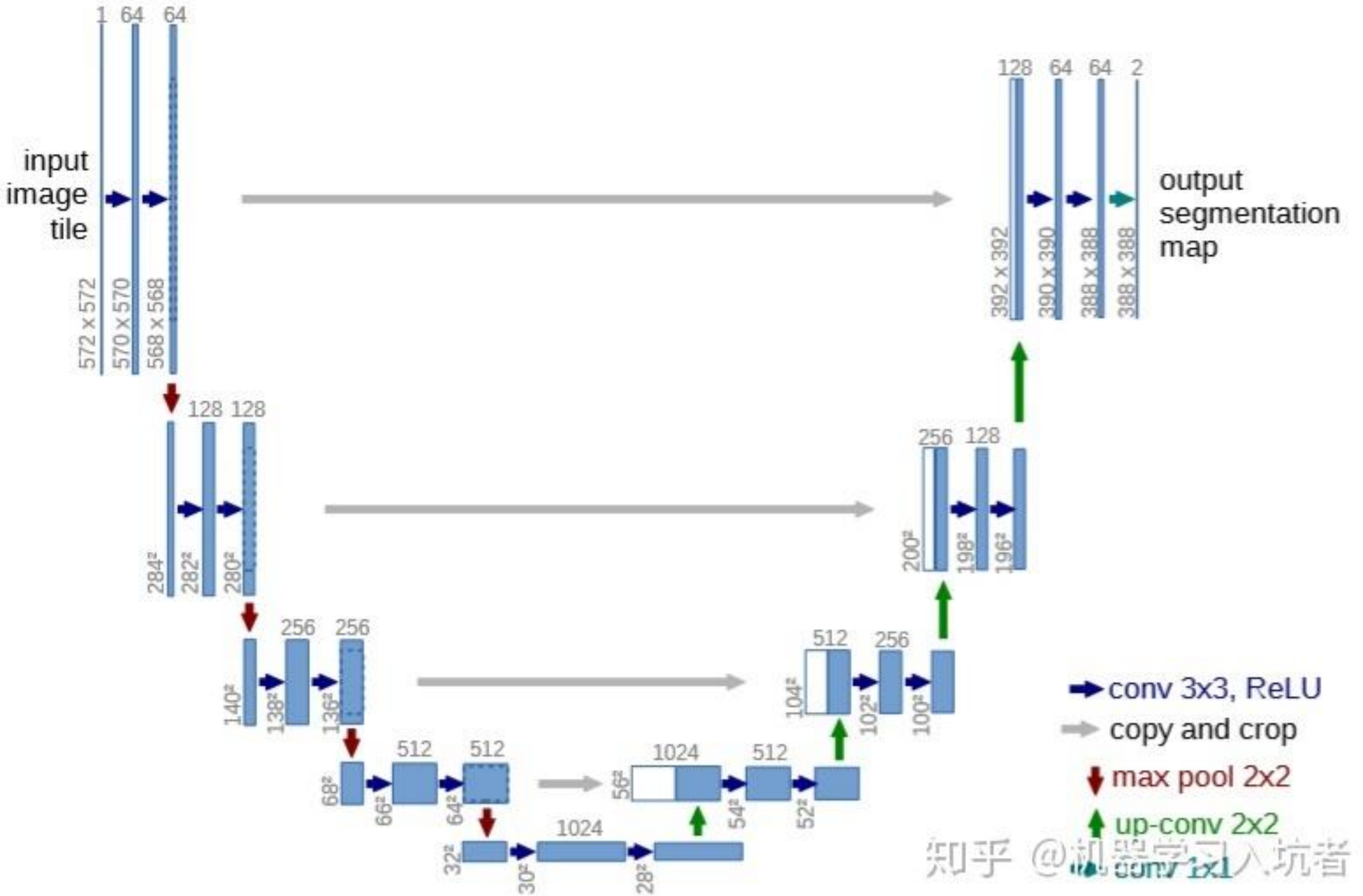
学习的知识

Table 1: Set of parameters considered for hyper-parameter search.

Parameters			
Learning rate	1e-3	1e-4	1e-5
Kernel	3	5	7
Filters	16, 32, 64	8, 16, 32, 32	8, 16, 16, 32, 32
Norm			
Batch	On	Off	
Weight	On	Off	

学习的知识

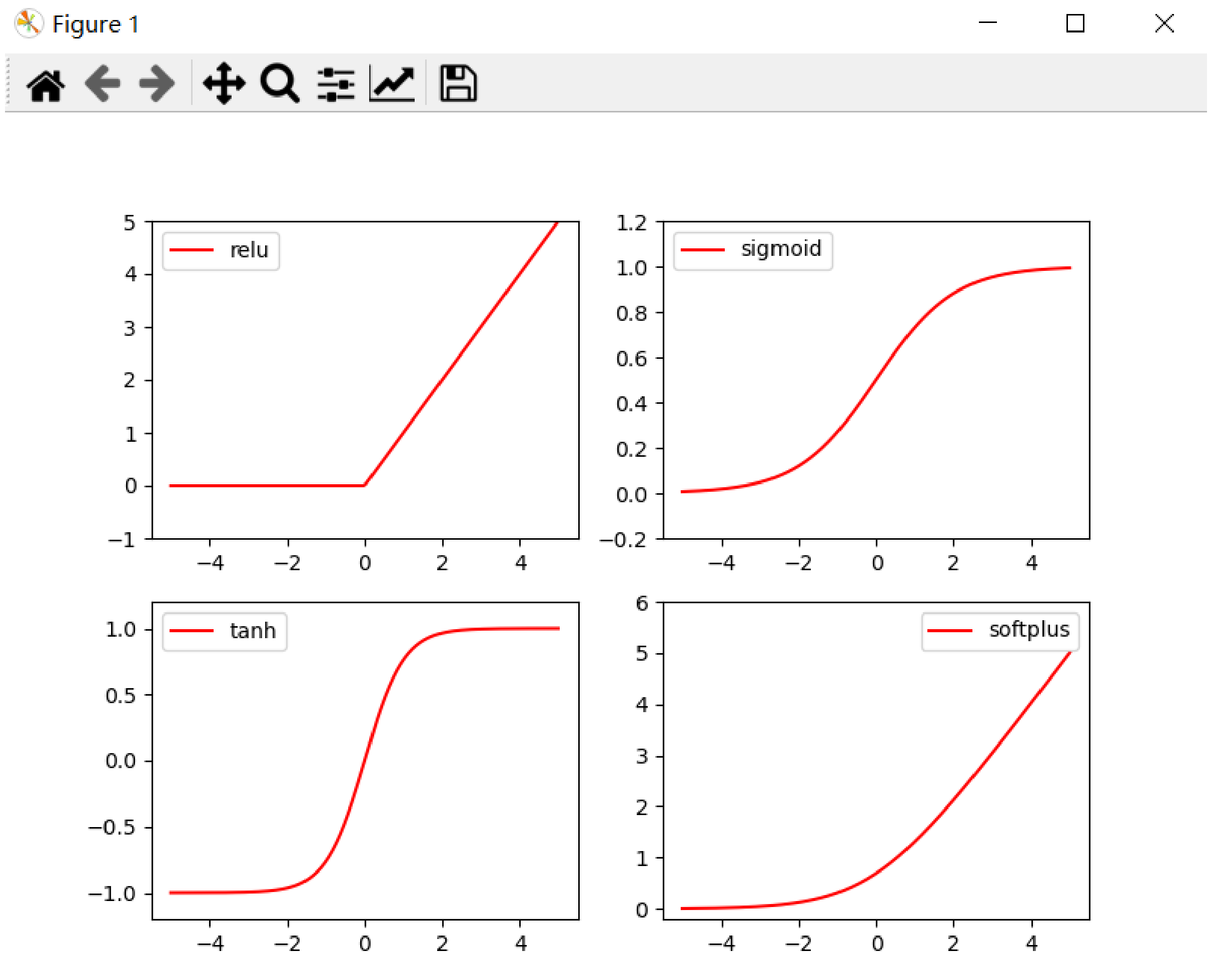
蓝色的矩形条表示特征图，矩形图上面的数字是通道数，
矩形图侧面的数字是x-y（特征图长和宽）



学习的知识1

ReLU

$$f(x)=\begin{cases}0, & x<0\\ x, & x\geq 0\end{cases}$$



学习的知识

卷积核的理解

卷积核在有的文档里也称为**过滤器（filter）：**

•每个卷积核具有长宽深三个维度;
•在某个卷积层中，可以有多个卷积核:
•下一层需要多少个feather map,
•本层就需要多少个卷积核。

转置卷积的推导

定义一个 **4×4 输入矩阵 input**:

$$input = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \\ x_6 & x_7 & x_8 & x_9 \\ x_{10} & x_{11} & x_{12} & x_{13} \\ x_{14} & x_{15} & x_{16} & x_{17} \end{bmatrix}$$

再定义一个 **3×3 标准卷积核 kernel**:

$$kernel = \begin{bmatrix} w_{0,0} & w_{0,1} & w_{0,2} \\ w_{1,0} & w_{1,1} & w_{1,2} \\ w_{2,0} & w_{2,1} & w_{2,2} \end{bmatrix}$$

设 步长 stride=1、填充 padding=0，则按 "valid" 卷积模式，可得 **2×2 输出矩阵 output**

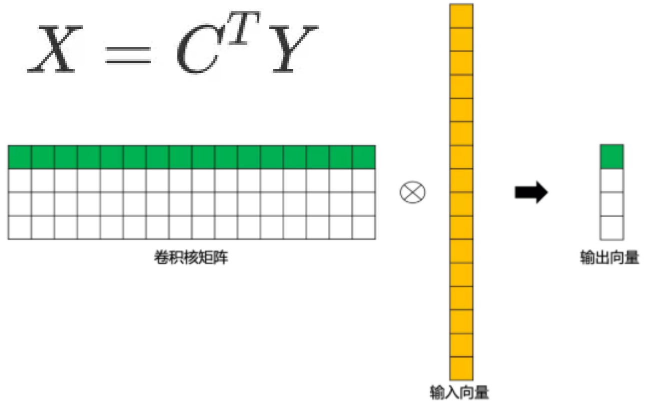
$$output = \begin{bmatrix} y_0 & y_1 \\ y_2 & y_3 \end{bmatrix}$$

这里，换一个表达方式，将输入矩阵 input 和输出矩阵 output 展开成 **16×1 列向量 X** 和 **4×1 列向量 Y**，可分别表示为:

接着，再用矩阵运算来描述标准卷积运算，设有 **新卷积核矩阵 C**:

$$X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \\ x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{15} \\ x_{16} \\ x_{17} \end{bmatrix}$$

$$X = C^T Y$$



$$Y = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix} \quad C = \begin{bmatrix} w_{0,0} & w_{0,1} & w_{0,2} & 0 & w_{1,0} & w_{1,1} & w_{1,2} & 0 & w_{2,0} & w_{2,1} & w_{2,2} & 0 & 0 & 0 & 0 & 0 \\ 0 & w_{0,0} & w_{0,1} & w_{0,2} & 0 & w_{1,0} & w_{1,1} & w_{1,2} & 0 & w_{2,0} & w_{2,1} & w_{2,2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & w_{0,0} & w_{0,1} & w_{0,2} & 0 & w_{1,0} & w_{1,1} & w_{1,2} & 0 & w_{2,0} & w_{2,1} & w_{2,2} & 0 \\ 0 & 0 & 0 & 0 & 0 & w_{0,0} & w_{0,1} & w_{0,2} & 0 & w_{1,0} & w_{1,1} & w_{1,2} & 0 & w_{2,0} & w_{2,1} & w_{2,2} \end{bmatrix}$$

学习的知识

而转置卷积（decoder的运算）其实就是要对这个过程进行逆运算，即 **通过 C 和 Y 得到 X**：

$$X = C^T Y$$

BN算法（Batch Normalization）

Patience（该参数设置但是未使用）：当patience个epoch过去而模型性能不提升时，学习率减少的动作会被触发

Python传参：*args 和 **kwargs 可以将不定数量的参数传递给函数：*args 和 **kwargs，前者适用于传入非键值对的可变数量的参数列表，后者适用于传入不定长度的键值对（字典），作为参数传递给函数。**kwargs 就是在*args 的基础上，添加了变量名以及转变了格式（哈希字典）

Paddle：with paddle.static.device_guard('gpu'):
等效pytorch：model = model.to(device)

学习的知识

PaddlePaddle DyGraph
是一个更加灵活易用的模式，可提供：
•更加灵活便捷的代码组织结构：使用python的执行控制流程和面向对象的模型设计
•更加便捷的调试功能：直接使用python的打印方法即时打印所需要的结果，从而检查正在运行的模型结果便于测试更改
•和静态执行图通用的模型代码：同样的模型代码可以使用更加便捷的DyGraph调试，执行，同时也支持使用原有的静态图模式执行
有关的动态图机制更多的实际模型示例请参考
[Paddle/models/dygraph]()

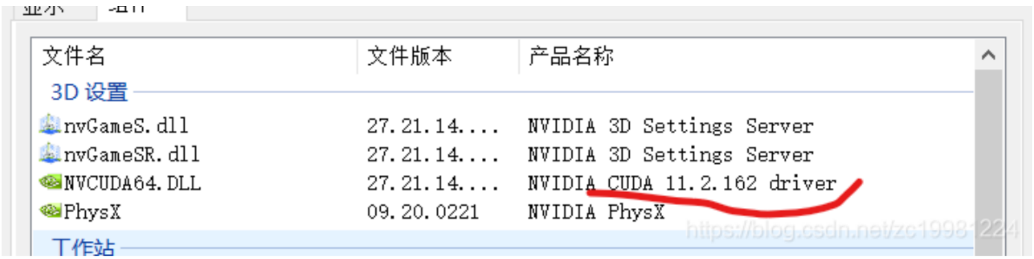
paddle.optimizer
Paddle在paddle.optimizer模块中实现了一些基于梯度的优化函数，包括梯度下降等常见的优化方法。在最小化网络损失值的过程中，需要先获取模型参数和学习率。

优化函数不会计算梯度，我们需要调用backward()来计算梯度。我们还需要在调用backward()函数之前调用optim.clear_grad()，原因是Paddle是默认梯度累加而不是梯度更新。

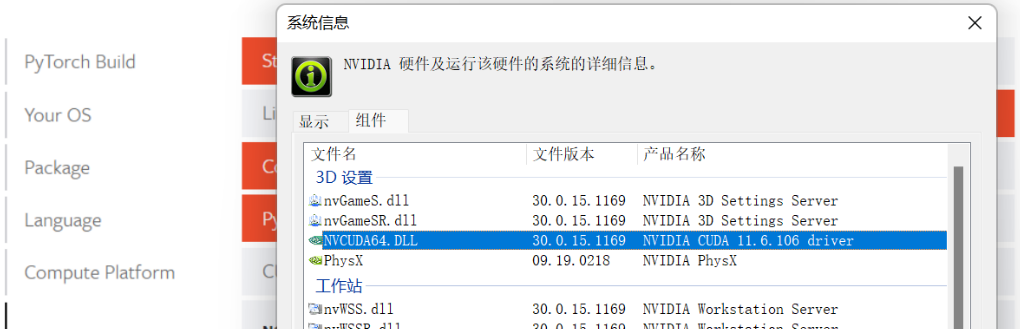
paddle.optimizer docs

正如我们所看到的这样，参数值是朝着正确的方向在更新的。

解决的问题1



打开pytorch官网https://pytorch.org/get-started/locally/根据版本需求选择适合的pytorch版本。如下图所示，这里一定要选择cuda11.1版本的，如果选择cuda10.2，结果就是在使用深度学习训练时导致不适配。（血与泪的教训）




```
C:\Windows\system32>python
Python 3.10.4 (tags/v3.10.4:9d38120, Mar 23 2022, 23:13:41) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import paddle
>>> paddle.utils.run_check()
Running verify PaddlePaddle program ...
W1120 17:15:03.425992 12544 gpu_resources.cc:61] Please NOTE: device: 0, GPU Compute Capability: 8.6, Driver API Version: 11.6, Runtime API Version: 11.6
W1120 17:15:03.438352 12544 gpu_resources.cc:91] device: 0, cuDNN Version: 8.4.
PaddlePaddle works well on 1 GPU.
PaddlePaddle works well on 1 GPUs.
PaddlePaddle is installed successfully! Let's start deep learning with PaddlePaddle now.
>>> A
```

解决的问题2

Previous train log deleted successfully
Backend TkAgg is interactive backend. Turning interactive mode on.
Epoch #1
Could not locate zlibwapi.dll. Please make sure it is in your library path!

3. Installing cuDNN on Windows

3.1. Prerequisites

For the latest compatibility software versions of the OS, CUDA, the CUDA driver, and the NVIDIA hardware, refer to the *NVIDIA cuDNN Support Matrix*.

3.1.1. Installing NVIDIA Graphic Drivers

Install up-to-date NVIDIA graphics drivers on your Windows system.

Procedure

1. Go to: NVIDIA download drivers
2. Select the GPU and OS version from the drop-down menus.
3. Download and install the NVIDIA driver as indicated on that web page. For more information, select the **ADDITIONAL INFORMATION** tab for step-by-step instructions for instal
4. Restart your system to ensure that the graphics driver takes effect.

3.1.2. Installing the CUDA Toolkit for Windows

Refer to the following instructions for installing CUDA on Windows, including the CUDA driver and toolkit: *NVIDIA CUDA Installation Guide for Windows*.

3.1.3. Installing Zlib

Zlib is a data compression software library that is needed by cuDNN.

Procedure

1. Download and extract the zlib package from ZLIB DLL. Users with a 32-bit machine should download the 32-bit ZLIB DLL.

   Note: If using Chrome, the file may not automatically download. If this happens, right-click the link and choose **Save link as...**. Then, paste the URL into a browser window.
2. Add the directory path of `zlibwapi.dll` to the environment variable PATH.

解决的问题3

Batch_size过大（源代码默认256）

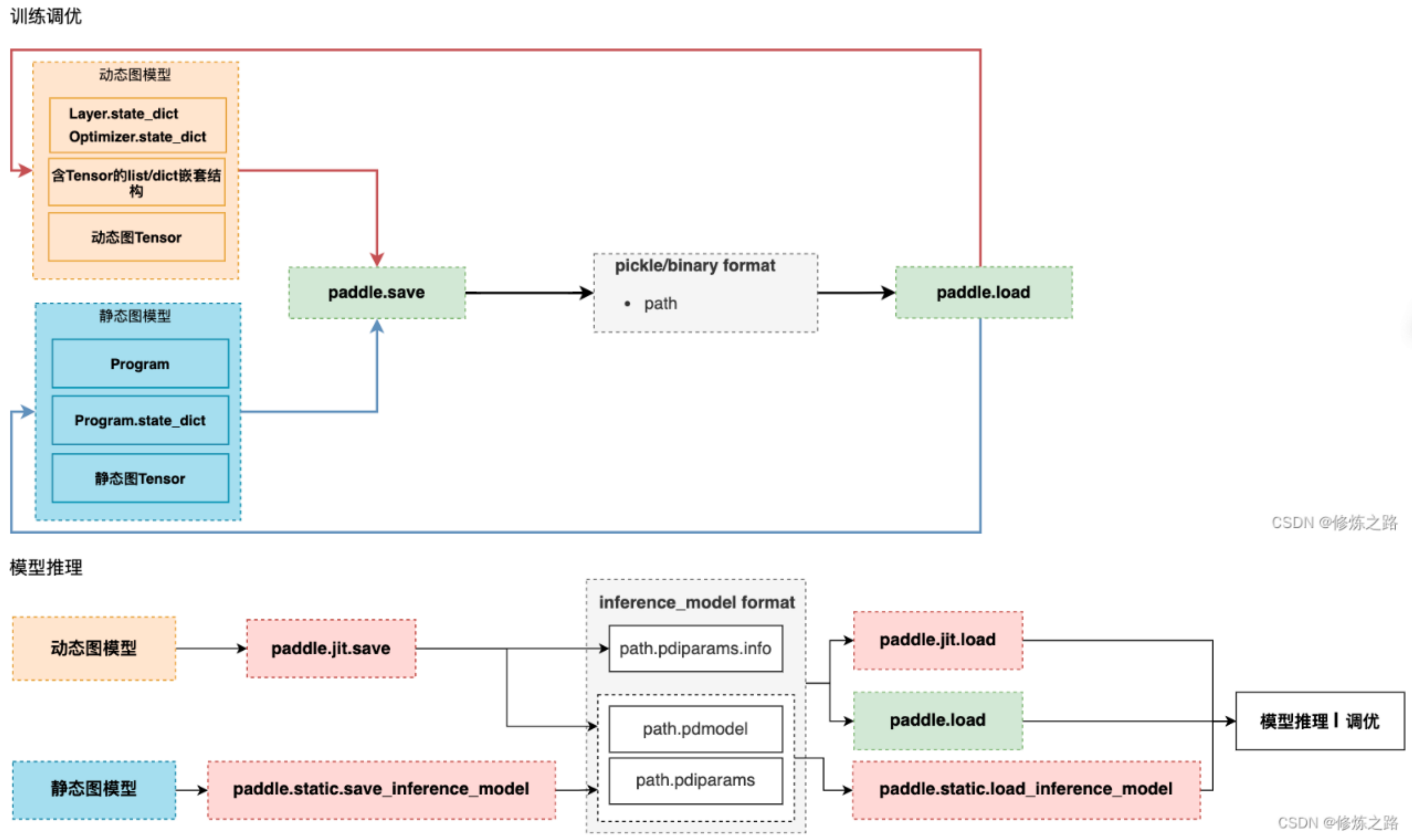
```
Traceback (most recent call last):
  File "D:\TEST\DeepCFD-master\DeepCFD_PY\train_functions.py", line 93, in train
    train_loss, train_metrics = epoch(scope, train_loader, on_train_batch, training=True)
  File "D:\TEST\DeepCFD-master\DeepCFD_PY\train_functions.py", line 39, in epoch
    loss, output = loss_func(model, tensors)
  File "D:/TEST/DeepCFD-master/DeepCFD_PY/main.py", line 29, in loss_func
    output = model(x)
  File "D:\Developer\Python\lib\site-packages\paddle\fluid\dygraph\layers.py", line 930, in __call__
    return self._dygraph_call_func(*inputs, **kwargs)
  File "D:\Developer\Python\lib\site-packages\paddle\fluid\dygraph\layers.py", line 915, in _dygraph_call_func
    outputs = self.forward(*inputs, **kwargs)
  File "D:\TEST\DeepCFD-master\DeepCFD_PY\models\UNetEx.py", line 118, in forward
    x = self.decode(x, tensors, indices, sizes)
  File "D:\TEST\DeepCFD-master\DeepCFD_PY\models\UNetEx.py", line 110, in decode
    x = paddle.concat([tensor, x], axis=1)
  File "D:\Developer\Python\lib\site-packages\paddle\tensor\manipulation.py", line 331, in concat
    return paddle.fluid.layers.concat(input=x, axis=axis, name=name)
  File "D:\Developer\Python\lib\site-packages\paddle\fluid\layers\tensor.py", line 343, in concat
    _C_ops.concat(input, out, 'axis', axis)
SystemError: (Fatal) Operator concat raises an struct paddle::memory::allocation::BadAlloc exception.
The exception content is
:ResourceExhaustedError:

Out of memory error on GPU 0. Cannot allocate 212.312500MB memory on GPU 0, 3.999573GB memory has been allocated and available memory is only 0.000000B.

Please check whether there is any other process using GPU 0.
1. If yes, please stop them, or start PaddlePaddle on another GPU.
2. If no, please decrease the batch size of your model.

 (at ..\paddle\fluid\memory\allocation\cuda_allocator.cc:87)
. (at ..\paddle\fluid\imperative\tracer.cc:307)
```

进程已结束,退出代码1

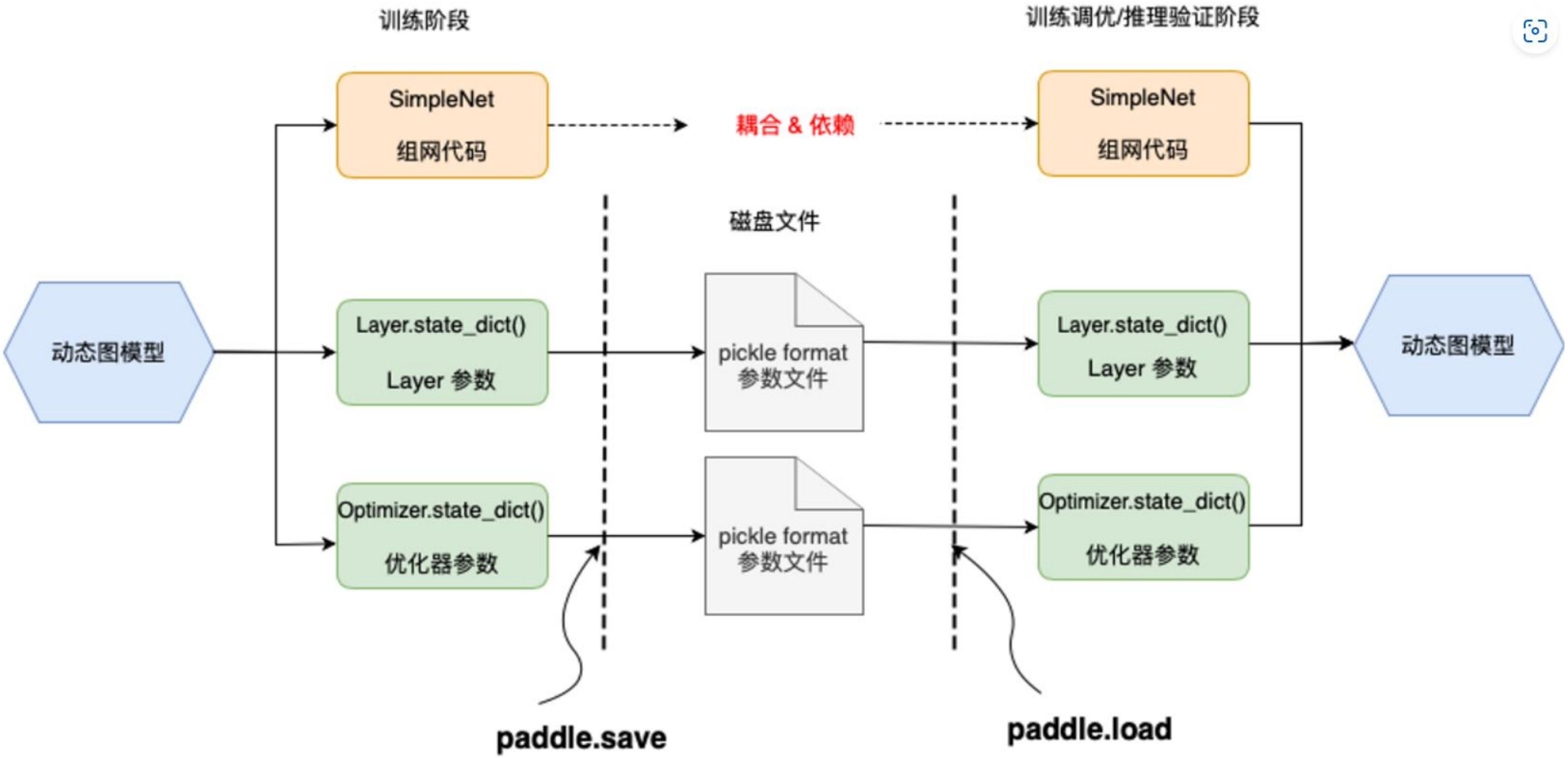
架构图1

模型保存和加载

PaddlePaddle 工作流



架构图2



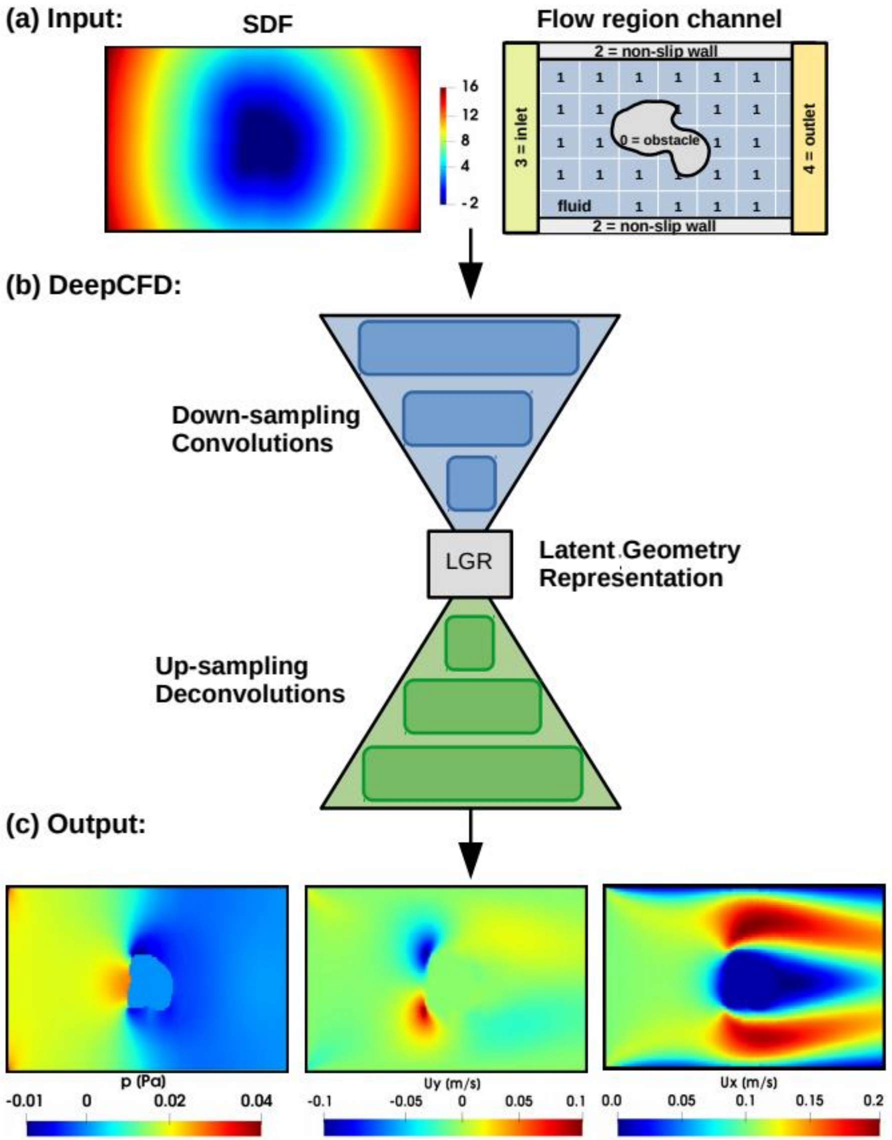
训练阶段
训练调优/推理验证阶段
SimpleNet
组网代码
耦合 & 依赖
SimpleNet
组网代码
磁盘文件
动态图模型
Layer.state_dict()
Layer 参数
pickle format
参数文件
Layer.state_dict()
Layer 参数
动态图模型
Optimizer.state_dict()
优化器参数
pickle format
参数文件
Optimizer.state_dict()
优化器参数
paddle.save
paddle.load

架构图3

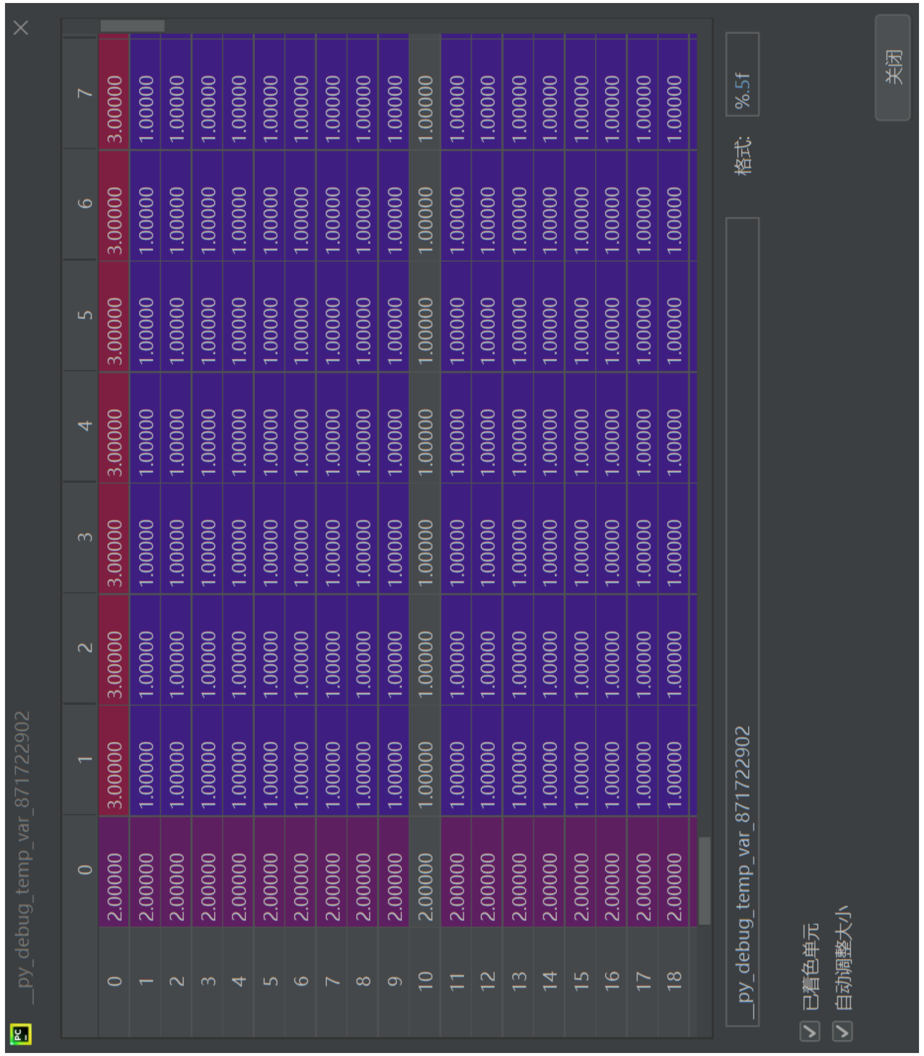
(a) Input channels with SDF and multi-class labeling of flow regions.

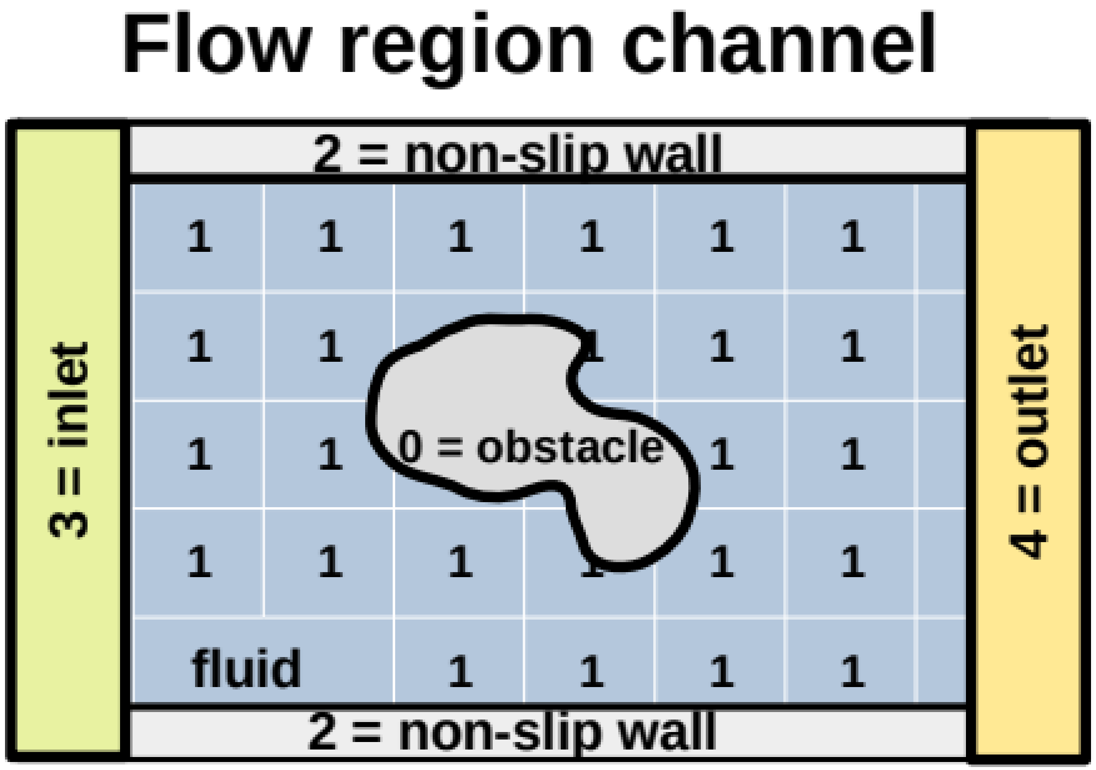
(b) Down-sampling convolutional operations create a latent representation of the flow geometry from the input.

(c) Up-sampling deconvolutions map the LGR to variables of interest

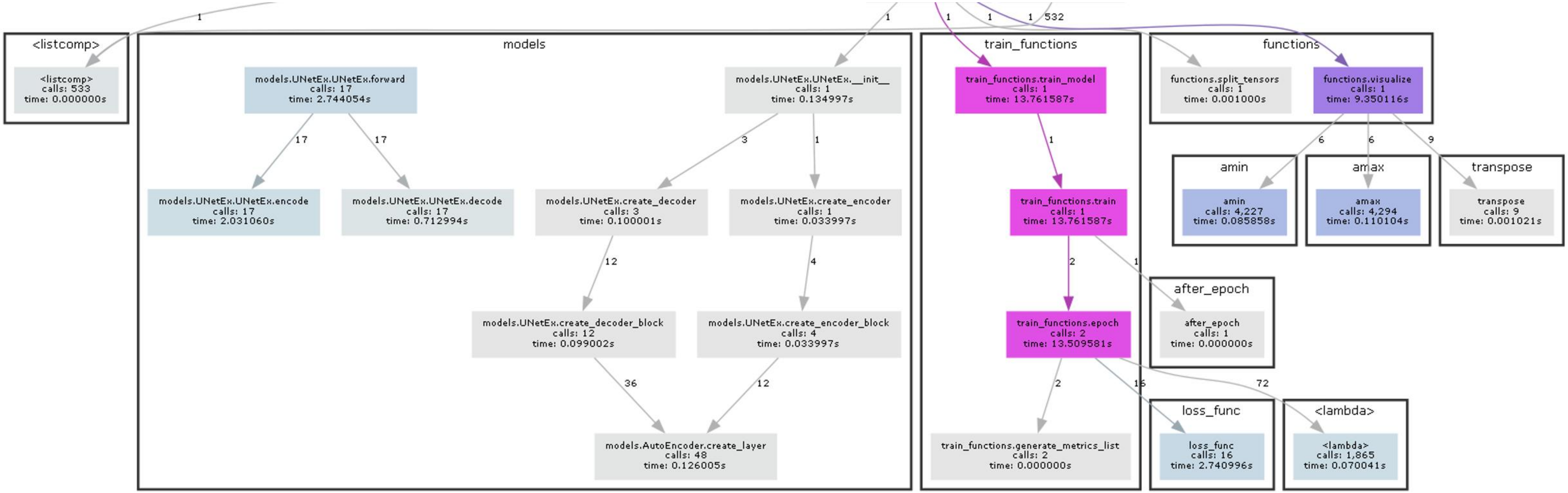


架构图4


_py_debug_temp_var_871722902
格式:
%.5f
关闭
已着色单元
自动调整大小


Flow region channel
2 = non-slip wall
3 = inlet
4 = outlet
0 = obstacle
fluid
1
2 = non-slip wall

架构5



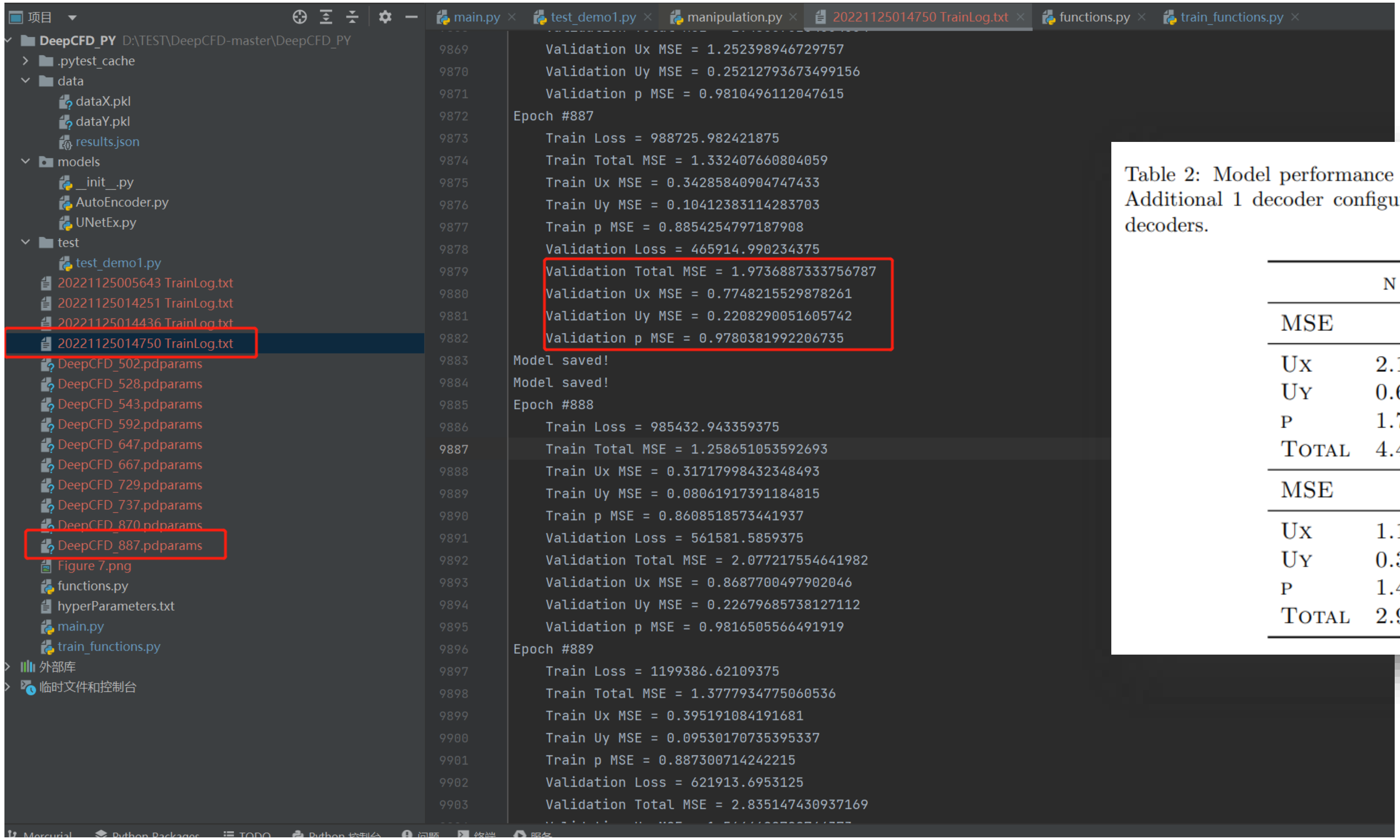
<listcomp>
<listcomp> calls: 533 time: 0.000000s
models
models.UNetEx.UNetEx.forward calls: 17 time: 2.744054s
models.UNetEx.UNetEx.__init__ calls: 1 time: 0.134997s
models.UNetEx.UNetEx.encode calls: 17 time: 2.031060s
models.UNetEx.UNetEx.decode calls: 17 time: 0.712994s
models.UNetEx.create_decoder calls: 3 time: 0.100001s
models.UNetEx.create_encoder calls: 1 time: 0.033997s
models.UNetEx.create_decoder_block calls: 12 time: 0.099002s
models.UNetEx.create_encoder_block calls: 4 time: 0.033997s
models.AutoEncoder.create_layer calls: 48 time: 0.126005s
train_functions
train_functions.train_model calls: 1 time: 13.761587s
train_functions.train calls: 1 time: 13.761587s
train_functions.epoch calls: 2 time: 13.509581s
train_functions.generate_metrics_list calls: 2 time: 0.000000s
functions
functions.split_tensors calls: 1 time: 0.001000s
functions.visualize calls: 1 time: 9.350116s
amin
amin calls: 4,227 time: 0.085858s
amax
amax calls: 4,294 time: 0.110104s
transpose
transpose calls: 9 time: 0.001021s
after_epoch
after_epoch calls: 1 time: 0.000000s
loss_func
loss_func calls: 16 time: 2.740996s
<lambda>
<lambda> calls: 1,865 time: 0.070041s

结果1

First, the model optimization procedure via hyper-parameter search is described, and the test error curves of DeepCFD are plotted against the ones of the baseline model

Furthermore, qualitative and quantitative analyses of the results are provided together with relevant discussion about the model accuracy and performance in comparison with the baseline [8] and with the standard CFD approach.

结果2



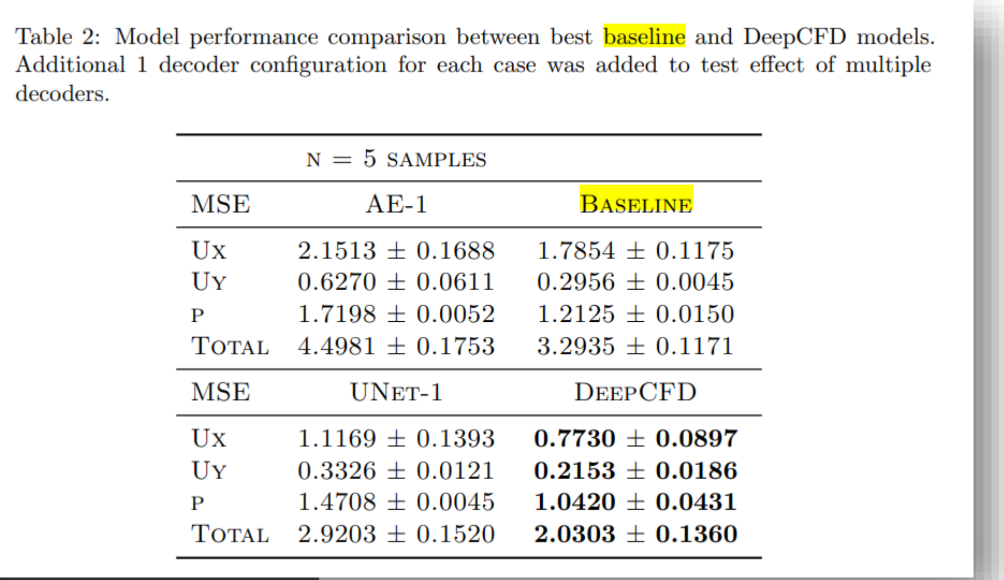
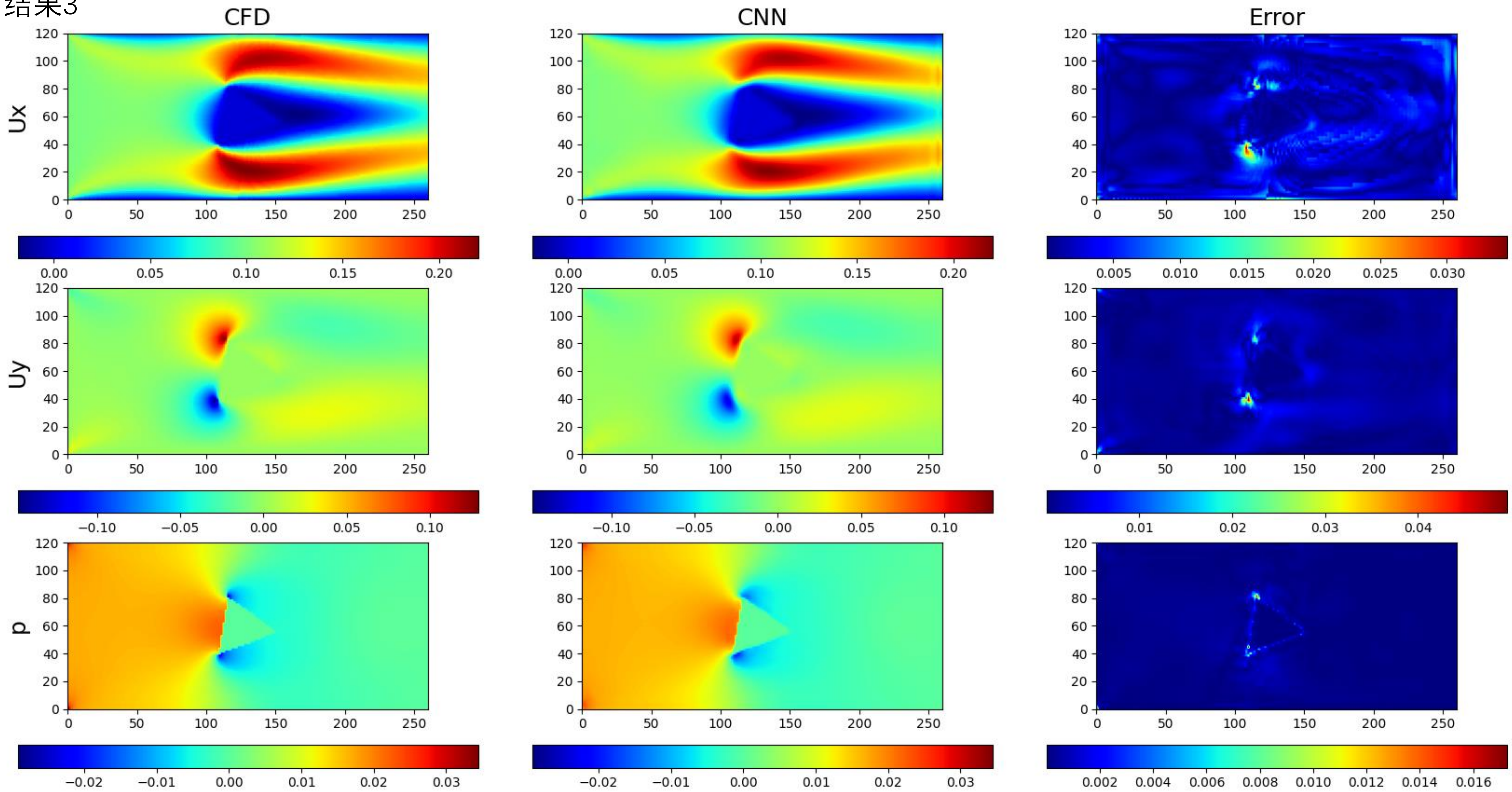


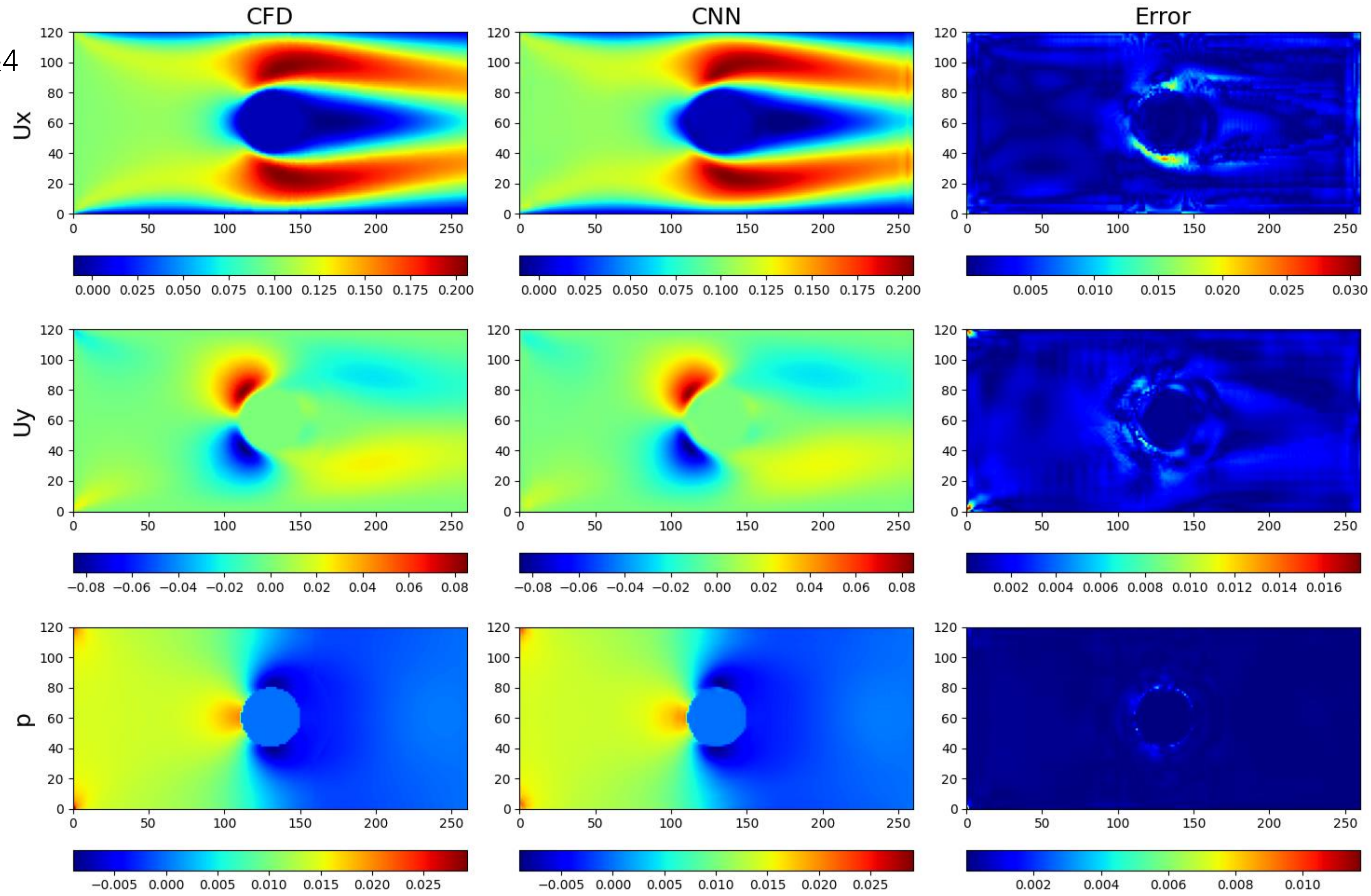
Table 2: Model performance comparison between best baseline and DeepCFD models. Additional 1 decoder configuration for each case was added to test effect of multiple decoders.

N = 5 SAMPLES		
MSE	AE-1	BASELINE
UX	2.1513 ± 0.1688	1.7854 ± 0.1175
UY	0.6270 ± 0.0611	0.2956 ± 0.0045
P	1.7198 ± 0.0052	1.2125 ± 0.0150
TOTAL	4.4981 ± 0.1753	3.2935 ± 0.1171
MSE	UNET-1	DEEPCFD
UX	1.1169 ± 0.1393	**0.7730 ± 0.0897**
UY	0.3326 ± 0.0121	**0.2153 ± 0.0186**
P	1.4708 ± 0.0045	**1.0420 ± 0.0431**
TOTAL	2.9203 ± 0.1520	**2.0303 ± 0.1360**

结果3

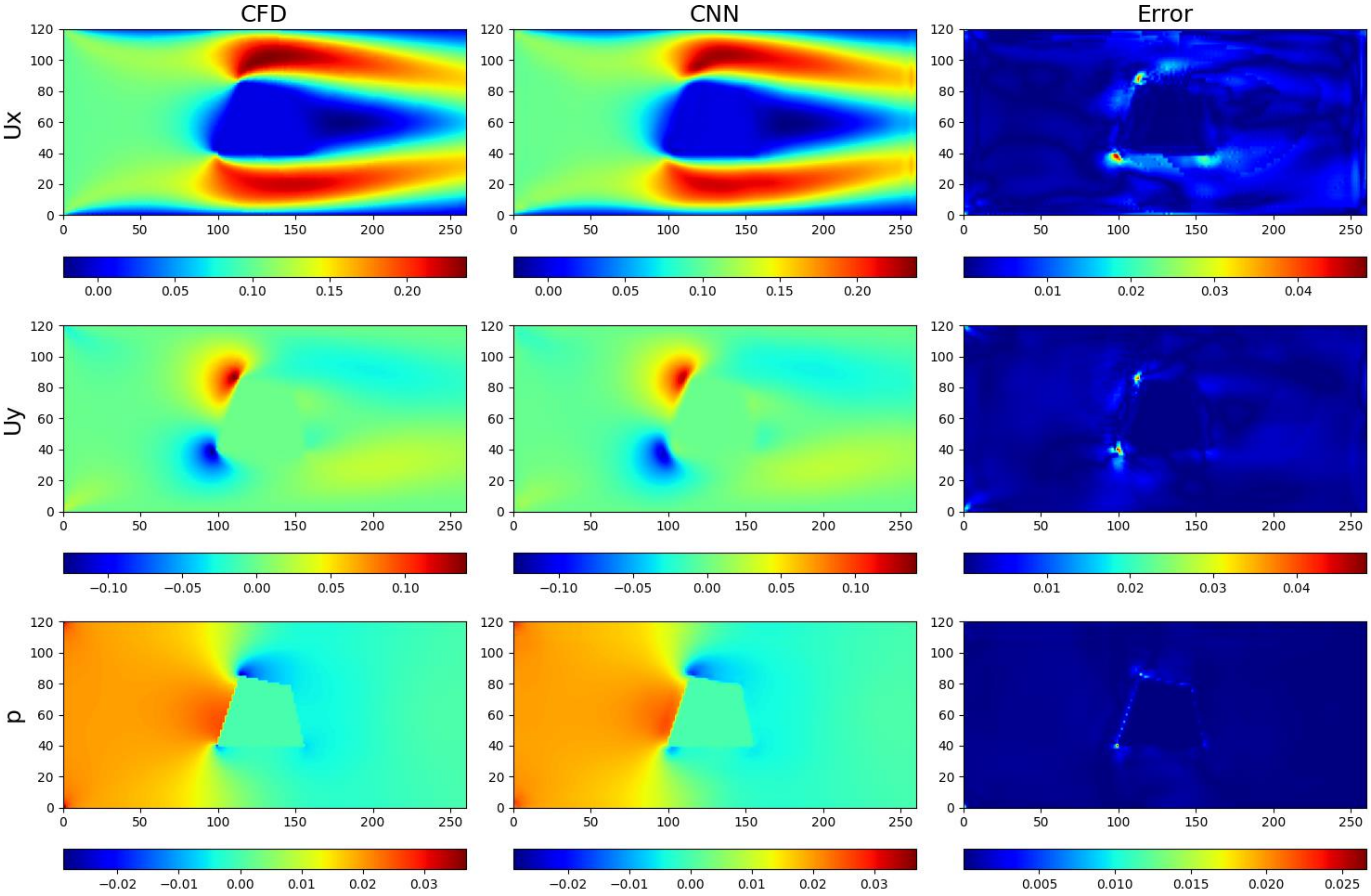


结果4



CFD
CNN
Error
Ux
Uy
p
0 50 100 150 200 250
0 20 40 60 80 100 120
0.000 0.025 0.050 0.075 0.100 0.125 0.150 0.175 0.200
0.005 0.010 0.015 0.020 0.025 0.030
−0.08 −0.06 −0.04 −0.02 0.00 0.02 0.04 0.06 0.08
0.002 0.004 0.006 0.008 0.010 0.012 0.014 0.016
−0.005 0.000 0.005 0.010 0.015 0.020 0.025
0.002 0.004 0.006 0.008 0.010

结果5



结果6

BaseLine出处 todo

X. Guo, W. Li, F. Iorio, Convolutional neural networks for steady flow approximation, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 16, Association for Computing Machinery, New York, NY, USA, 2016, p. 481–490.

Test Error(MSE)