# 0、前置准备

需要根据《HW_6作业思路提示》，完成一些初步的工作。

# 1、进行 fp16 加速并测试速度

及格标准：设置build_config，对模型进行fp16优化；

**思路提示**：认真阅读老师提供的代码，修改builder.sh中的输入参数。

优秀标准：编写fp16 版本的layer_norm算子，使模型最后运行fp16版本的layer_norm算子。

**思路提示**：layer_norm 核函数的实现时，使用函数模板来处理，通过将类型作为参数传递给模板，可使编译器生成该类型的函数。

# 2、进行 int8 加速并测试速度

完善calibrator.py内的todo函数，使用calibrator_data.txt 校准集，对模型进行int8量化加速。

**思路提示**：将FP32降为INT8的过程相当于信息再编码（re-encoding information），就是原来使用32bit来表示一个tensor，现在使用8bit来表示一个tensor，还要求精度不能下降太多。

这部分整体还是基本的操作，主要完成TODO部分工作即可。

首先是读入calibrator_data.txt文件，转化为模型的输入数据。

```
def text2inputs(tokenizer, text):
    encoded_input = tokenizer.encode_plus(text, return_tensors = "pt")

    input_ids = encoded_input['input_ids'].int().detach().numpy()
    token_type_ids = encoded_input['token_type_ids'].int().detach().numpy()
    # position_ids = torch.arange(0, encoded_input['input_ids'].shape[1]).int().view(1, -1).numpy()
    seq_len = encoded_input['input_ids'].shape[1]
    position_ids = np.arange(seq_len, dtype = np.int32).reshape(1, -1)
    input_list = [input_ids, token_type_ids, position_ids]

    return input_list

# TODO: your code, read inputs
with open(data_txt, "r") as f:
            lines = f.readlines()
```

```
            for i in range(0, num_inputs):
                inputs = text2inputs(tokenizer, lines[i])
                self.input_ids_list.append(inputs[0])
                self.token_type_ids_list.append(inputs[1])
                self.position_ids_list.append(inputs[2])
                if i % 10 == 0:
                    print("text2inputs:" + lines[i])
```

然后是get_batch()中的代码补全，这里贴出参考实现代码，如下：

```
# TODO your code, copy input from cpu to gpu
input_ids = self.input_ids_list[self.current_index]
token_type_ids = self.token_type_ids_list[self.current_index]
position_ids = self.position_ids_list[self.current_index]

seq_len = input_ids.shape[1]
if seq_len > self.max_seq_length:
    print(seq_len)
    print(input_ids.shape)
    input_ids = input_ids[:, :self.max_seq_length]
    token_type_ids = token_type_ids[:, :self.max_seq_length]
    position_ids = position_ids[:, :self.max_seq_length]
    print(input_ids.shape)

cuda.memcpy_htod(self.device_inputs[0], input_ids.ravel())
cuda.memcpy_htod(self.device_inputs[1], token_type_ids.ravel())
cuda.memcpy_htod(self.device_inputs[2], position_ids.ravel())

self.current_index += self.batch_size
```

参考资料：

- https://github.com/NVIDIA/TensorRT （TensorRT GitHub，特别是demo 和samples部分）
- https://docs.nvidia.com/deeplearning/tensorrt/index.html （TensorRT 官方文档）