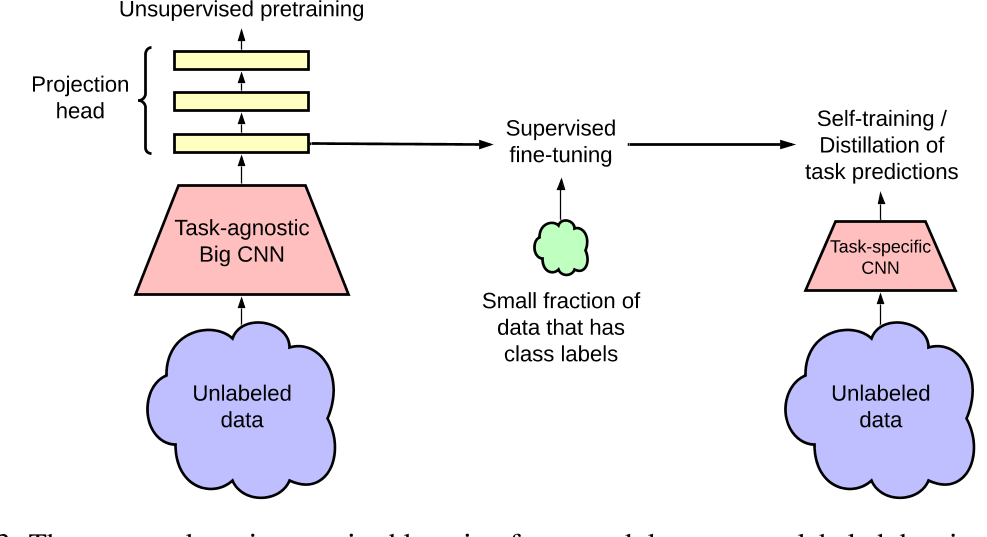
**Simclrv2:**

1. **模型背景**

Googlebrain的研究人员在早期开发的SimCLR框架之上提出了SimCLRv2，SimCLR框架使用（一个简单的视觉表征对比学习框架）进行无监督的预训练。而SimCLRv2使用了一种独特的方法，包括使用的无监督预训练、有监督的微调和未标记数据的蒸馏，这种方法将ImageNet数据集上SOTA结果提高21.6%。模型结构如下图：



1. **导出onnx模型**

训练原始的pytorch模型，保存权重文件，在torch文件夹中有训练得到的.pth，运行python divert.py，需要输入权重文件路径以及保存onnx模型名称，得到转换后的onnx模型。

1. **转换bmodel**

输入以及simclr2.qtable在scripts文件夹中

**Batch\_size = 1**

model\_transform.py \

--model\_name simclrv2\_1b \

--model\_def ../model\_simclr.onnx \

--input\_shapes [[1,3,32,32]] \

--keep\_aspect\_ratio \

--pixel\_format rgb \

--mlir simclrv2\_1b.mlir \

--test\_input 9\_3157.jpg \

--test\_result simclrv2\_top\_outputs.npz

**Fp16:**

model\_deploy.py \

--mlir simclrv2\_1b.mlir \

--quantize F16 \

--chip bm1684x \

--test\_input simclrv2\_1b\_in\_f32.npz \

--test\_reference simclrv2\_top\_outputs.npz \

--tolerance 0.99,0.99 \

--model simclrv2\_fp16\_1b.bmodel

**Fp32:**

model\_deploy.py \

--mlir simclrv2\_1b.mlir \

--quantize F32 \

--chip bm1684x \

--test\_input simclrv2\_1b\_in\_f32.npz \

--test\_reference simclrv2\_top\_outputs.npz \

--tolerance 0.99,0.99 \

--model simclrv2\_fp32\_1b.bmodel

**Int8:**

run\_calibration.py simclrv2\_1b.mlir \

--dataset test/ \

--input\_num 100 \

-o simclrv2\_cali\_table

model\_deploy.py \

--mlir simclrv2\_1b.mlir \

--quantize INT8 \

--chip bm1684x \

--calibration\_table simclrv2\_cali\_table \

--test\_input simclrv2\_1b\_in\_f32.npz \

--test\_reference simclrv2\_top\_outputs.npz \

--quantize\_table simclr2.qtable \

--model simclrv2\_int8\_1b.bmodel

**Batch\_size = 4**

model\_transform.py \

--model\_name simclrv2\_4b \

--model\_def ../model\_simclr.onnx \

--input\_shapes [[4,3,32,32]] \

--keep\_aspect\_ratio \

--pixel\_format rgb \

--test\_input images\_data.npz \

--test\_result simclrv2\_top\_outputs.npz \

--mlir simclrv2\_4b.mlir

**Fp16:**

model\_deploy.py \

--mlir simclrv2\_4b.mlir \

--quantize F16 \

--chip bm1684x \

--test\_input simclrv2\_4b\_in\_f32.npz \

--test\_reference simclrv2\_top\_outputs.npz \

--tolerance 0.99,0.99 \

--model simclrv2\_fp16\_4b.bmodel

**Fp32:**

model\_deploy.py \

--mlir simclrv2\_4b.mlir \

--quantize F32 \

--chip bm1684x \

--test\_input simclrv2\_4b\_in\_f32.npz \

--test\_reference simclrv2\_top\_outputs.npz \

--tolerance 0.99,0.99 \

--model simclrv2\_fp32\_4b.bmodel

**Int8:**

run\_calibration.py simclrv2\_4b.mlir \

--dataset test/ \

--input\_num 100 \

-o simclrv2\_cali\_table

model\_deploy.py \

--mlir simclrv2\_4b.mlir \

--quantize INT8 \

--chip bm1684x \

--calibration\_table simclrv2\_cali\_table \

--test\_input simclrv2\_4b\_in\_f32.npz \

--test\_reference simclrv2\_top\_outputs.npz \

--quantize\_table simclr2.qtable \

--model simclrv2\_int8\_4b.bmodel

1. **精度测试及结果展示**

测试代码在python文件夹中,测试数据集test\_batch，可参考README.md，得到分类结果的.json文件。

python3 simclrv2.py --input test\_batch --bmodel models/simclrv2\_fp32\_1b.bmodel --dev\_id 0

结果如下图：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试平台 | 测试程序 | 测试模型 | preprocess\_time | inference\_time | postprocess\_time |
| BM1684X SoC | simclrv2.py | simclrv2\_fp16\_1b.bmodel | 0.14 | 2.07 | 0.17 |
| BM1684X SoC | simclrv2.py | simclrv2\_fp32\_1b.bmodel | 0.15 | 3.98 | 0.17 |
| BM1684X SoC | simclrv2.py | simclrv2\_int8\_1b.bmodel | 0.15 | 1.85 | 0.18 |
| BM1684X SoC | simclrv2.py | simclrv2\_int8\_4b.bmodel | 0.10 | 0.81 | 0.11 |

1. **mac利用率测试结果**

模型精度测试代码在model\_eval文件夹中，pt\_eval.py对原始模型进行测试，onnx\_eval.py对onnx的精度进行测试，bmodel\_eval.py测试f32，f16以及int8bmodel的精度，bmodel测试需要导入结果的.json文件。测试结果如下：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 模型 | pt | onnx | F32 | F16 | Int8 |
| acc | 89.27% | 89.26% | 89.26% | 89.25% | 89.23% |
| mac\_utilization, |  |  | 50.05% | 16.74% | 9.77% |

1. **实验分析**

转换int8模型时遇到问题，由于模型中relu层太多，转换int8时前面几层和后面几层精度损失比较大，在qtable加了一些网络前侧和网络后面使用fp16的办法提升了模型的精度，之后可以尝试将原始模型relu替换为其他函数，测试原始模型的性能以及转换的bmodel的性能。

1. **可视化展示**

转换onnx模型以及各个模型的精度测试均在notebook文件夹.ipynp中。