D1 开发板 YOLO 算法优化

1.实验目的

熟悉一个 RISC-V 架构的硬件平台，并在之上启动一个 Linux 操作系统。然后在此之上运行 YOLO 算法，之后需要对卷积层进行优化，在此过程会使用嵌入的汇编代码进行性能监测，并对优化的性能进行分析，在这个过程中我们可以把书中学习到的缓存原理知识应用到实际的工程开发中，并真切地感受到其发挥的作用。最后通过 HHB 流程对 RISC-V 的向量扩展的应用有一个初步的认识。

2.实验步骤

（1）将修改过的 Tina Linux 固件烧写到 D1 开发板中

（2）在 D1 开发板上运行 hello_world 的全流程

```
C:\Users\Dong-Yuxuan\Desktop\Documents>adb push C:\Users\Dong-YuXuan\Desktop\Documents\hello_world /
C:\Users\Dong-YuXuan\Desktop\Documents\hello_world: 1 file pushed. 4.2 MB/s (3575576 bytes in 0.813s)

C:\Users\Dong-Yuxuan\Desktop\Documents>adb shell


BusyBox v1.27.2 () built-in shell (ash)

                         Tina is Based on OpenWrt!
 ----------------------------------------------
 Tina Linux (Neptune, 5C1C9C53)
 ----------------------------------------------
root@TinaLinux:/# ls
base                        riscv64-unknown-linux-gnu
bin                         rom
dev                         root
etc                         sbin
hello_world                 spec
lib                         stress
lib64xthead                 sys
mnt                         tmp
overlay                     usr
proc                        var
rdinit                      www
root@TinaLinux:/# chmod +x hello_world
root@TinaLinux:/# hello_world
/bin/sh: hello_world: not found
root@TinaLinux:/# ./hello_world
Hello NeZha
root@TinaLinux:/#
```

（3）撰写带有缓存优化版本的卷积函数，并通过 qemu 的测试（需要写出优化思路，以及优化的原理）

Qemu 的测试结果：

```
data/dog.jpg: Predicted in 22.908990 seconds.
dog: 57%
car: 52%
truck: 56%
car: 62%
bicycle: 59%
```

优化思路与原理：

由于目前对 C 语言的缓存优化方法知识有限，了解到对 C 语言缓存优化的方法有减少全局变量，增强数据访问的局部性（通过数据重排，或者改变多重循环次序等来实现）以及对载入数据尽可能地复用来实现，故从这些方面进行入手：（均基于 base 版本的代码进行调整）

对循环次序进行调整：由于 base 版本的代码循环次序已经是从外到内，故调整后程序运行时间略有变长；

将 i,j,k 和 c_i,c_j,c_k 的三个循环分别改写成一个循环加三个表达式

定义循环变量的形式：由于每次都要重复计算循环变量，而且没有根本上解决访存问题，故程序运行时间明显变长；

将常量表达式用一个变量事先定义，免去后续反复读取与计算：程序运行时间稍稍缩短（最终采用了这种方法）

（4）在 D1 开发板上进行三个版本的 YOLO 性能测试，并撰写分析报告

	Base	Gemm	Cache-opt
Cycles	1916034900	406006793	1916735286
Instructions	1290638893	311071250	1290626918
L1D cache read access	355812333	88712921	355804309
L1D cache read miss	3188099	161956	3189453
L1D cache read hit ratio	0.991040	0.998174	0.991036
L1D cache write access	1640093	44567687	1637959
L1D cache write miss	63189	11974	62872
L1D cache write hit ratio	0.961472	0.999731	0.961616
执行时间（s）	143.748504	27.880296	143.903379

对于 gemm 版本的分析可以看出：gemm_nn 等 4 个函数中使用了 #pragma omp parallel for 以多线程的方式进行运行；此外，im2col 函数中使用了 get_pixel，省去了 padding 的过程，同时数据重排卷积运算的参数，使得这种算法在多个卷积核运算的场景下对运算加速十分有效

（5）使用平头哥 HHB 流程编译 mobilenet 生成两个版本的可执行

文件（一个带向量扩展，一个不带扩展），并分别在 qemu 和 D1 开发板运行。另行选取三张图片重复上述过程，并将实验结果统计在如下表格中。

下图为使用示例的 cat.jpg 作为输入，在 qemu 上运行的结果：

```
root@dd4950e18611:/home/example/c906/caffe_mobilenetv1# qemu-riscv64 -cpu c906fd
v c_runtime_c906 model.params data.0.bin
Run graph execution time: 23988.98047ms, FPS=0.04

=== tensor info ===
shape: 1 3 224 224
data pointer: 0x40010a3010

=== tensor info ===
shape: 1 1000 1 1
data pointer: 0x34d880
The max_value of output: 0.164185
The min_value of output: 0.000000
The mean_value of output: 0.000987
The std_value of output: 0.000087
 ------------ top5: ------------
277(red fox, Vulpes vulpes): 0.164185
282(tiger cat): 0.160278
278(kit fox, Vulpes macrotis): 0.143188
285(Egyptian cat): 0.088562
281(tabby, tabby cat): 0.049103
root@dd4950e18611:/home/example/c906/caffe_mobilenetv1# qemu-riscv64 -cpu c906fd
v c_runtime_ref model.params data.0.bin
Run graph execution time: 24304.46875ms, FPS=0.04

=== tensor info ===
shape: 1 3 224 224
data pointer: 0x40010a3010

=== tensor info ===
shape: 1 1000 1 1
data pointer: 0x306750
The max_value of output: 0.164185
The min_value of output: 0.000000
The mean_value of output: 0.000987
The std_value of output: 0.000087
 ------------ top5: ------------
277(red fox, Vulpes vulpes): 0.164185
282(tiger cat): 0.160278
278(kit fox, Vulpes macrotis): 0.143188
285(Egyptian cat): 0.088562
281(tabby, tabby cat): 0.049103
root@dd4950e18611:/home/example/c906/caffe_mobilenetv1#
```

测试了三张图片，图 1 为老虎(tiger)，图 2 为斑马(zebra)，图 3 为猫头鹰(owl)

表 1 在不同平台下不同输入的执行时间（ms）

	Qemu_ref	Qemu_c906	D1_ref	D1_c906

图片 1	24963.88086	7661.69580	41118.89453	358.31845
图片 2	24975.73242	7432.19141	41212.12891	358.79007
图片 3	24773.45703	7851.60059	41193.26562	358.48166

表 2 在不同平台下不同输入的分类结果

	Qemu_ref	Qemu_c906	D1_ref	D1_c906
图片 1	292(tiger, Panthera tigris): 0.926758 282(tiger cat): 0.050690 340(zebra): 0.011665 43(frilled lizard, Chlamydosaurus kingi): 0.001449 290(jaguar, panther, Panthera onca, Felis onca): 0.001019	292(tiger, Panthera tigris): 0.935059 282(tiger cat): 0.044769 340(zebra): 0.011955 43(frilled lizard, Chlamydosaurus kingi): 0.001420 290(jaguar, panther, Panthera onca, Felis onca): 0.001045	292(tiger, Panthera tigris): 0.926758 282(tiger cat): 0.050690 340(zebra): 0.011665 43(frilled lizard, Chlamydosaurus kingi): 0.001449 290(jaguar, panther, Panthera onca, Felis onca): 0.001019	292(tiger, Panthera tigris): 0.934082 282(tiger cat): 0.045776 340(zebra): 0.011856 43(frilled lizard, Chlamydosaurus kingi): 0.001452 290(jaguar, panther, Panthera onca, Felis onca): 0.001060
图片 2	340(zebra): 0.995117 292(tiger, Panthera tigris): 0.001394 396(lionfish): 0.000798 282(tiger cat): 0.000674 83(prairie chicken, prairie grouse, prairie fowl): 0.000390	340(zebra): 0.996094 292(tiger, Panthera tigris): 0.001251 396(lionfish): 0.000686 282(tiger cat): 0.000606 83(prairie chicken, prairie grouse, prairie fowl): 0.000363	340(zebra): 0.995117 292(tiger, Panthera tigris): 0.001394 396(lionfish): 0.000798 282(tiger cat): 0.000674 83(prairie chicken, prairie grouse, prairie fowl): 0.000390	340(zebra): 0.996094 292(tiger, Panthera tigris): 0.001281 396(lionfish): 0.000673 282(tiger cat): 0.000621 83(prairie chicken, prairie grouse, prairie fowl): 0.000363
图片 3	24(great grey owl, great gray owl, Strix nebulosa): 0.854004 384(indri, indris, Indri indri, Indri brevicaudatus): 0.017715 383(Madagascar cat, ring-tailed lemur, Lemur catta): 0.009262 282(tiger cat): 0.009155 142(dowitcher): 0.008469	24(great grey owl, great gray owl, Strix nebulosa): 0.865723 384(indri, indris, Indri indri, Indri brevicaudatus): 0.016296 383(Madagascar cat, ring-tailed lemur, Lemur catta): 0.009338 282(tiger cat): 0.008942 142(dowitcher): 0.008736	24(great grey owl, great gray owl, Strix nebulosa): 0.854004 384(indri, indris, Indri indri, Indri brevicaudatus): 0.017715 383(Madagascar cat, ring-tailed lemur, Lemur catta): 0.009262 282(tiger cat): 0.009155 142(dowitcher): 0.008469	24(great grey owl, great gray owl, Strix nebulosa): 0.864258 384(indri, indris, Indri indri, Indri brevicaudatus): 0.016846 383(Madagascar cat, ring-tailed lemur, Lemur catta): 0.009438 282(tiger cat): 0.009033 142(dowitcher): 0.008759

可以看到，在板载 c906 处理器的情况下，使用向量扩展的版本比不使用扩展的版本运行速度快数十倍，从而充分发挥出开发板的优势；而即便在不带 c906 处理器的 qemu 上，向量拓展版本还是比对照版本有一定速度上的优势

3.实验分析与总结

本实验分别通过 qemu 搭载的 linux 系统和 D1 开发板搭载的 linux 系统运行了 YOLO 算法，并在已有的缓存知识的基础上，通过修改 C 代码实现了一定的缓存优化。最后通过 HHB 工具实现了 MobileNet 模型的 C 语言转化，链接与执行，初步了解了 RISC-V 的向量扩展

4.实验收获

本实验最大的感触在于板载 linux 系统的使用与缓存优化的实现，搭建了软件与硬件连接的桥梁，也深刻认识到目前所学的体系架构知识是如何为工程开发的性能优化服务的