HLS for Human Activity Recognition using CNN in Keras

Group Number: 2

Group Members :

Abhinit Singh - 234101058

Akshat Dubey - 234101059

Param Bharatbhai Patel - 234101062

Vinayak Bhausaheb Ichake - 234101066

Prince Kumar - 210101081

1. Description of the model :

Task of model	Number of layers	Type of Layers	Count	Model size
Human Activity Recognition	7	Input Layer Conv2D MaxPooling2D Reshape Dense Layer	1 1 1 1 3	8.9 MB

Topology:

Layer name	Layer type	Input shape	Output shape
Conv2d_1_input	InputLayer	[[None,90,3,1]]	[None,90,3,1]
Conv2d_1	Conv2D	[[None,90,3,1]]	[None,89,2,128]
max_pooling2d_1	MaxPooling2D	[[None,89,2,128]]	[None,44,1,128]
faltten_1	Reshape	[[None,44,1,128]]	[None,5632]
dense_1	Dense	[[None,5632]]	[None,128]
dense_2	Dense	[[None,128]]	[None,128]
dense_3	Dense	[[None,128]]	[None,6]

2**. Changes made to make keras2c generated files synthesizable and description of the change made.**

❖ **Simulation and Synthesis:**

- Resolved linkage errors : Various functions defined in include folder of keras2c in our converted code. So, we included those headers for resolving linkage errors.

```
#include "k2c_include.h"
#include "k2c_tensor_include.h"
#include "k2c_pooling_layers.h"
#include "k2c_convolution_layers.h"
#include "k2c_core_layers.h"
#include "k2c_helper_functions.h"
#include "k2c_recurrent_layers.h"
#include "k2c_activations.h"
```

- Resolved redefinition errors : In multiple 'for' loops, we were getting redefinition errors due to same definition of loop iterators in each loop. So, we declared that variable once and update for each loop respectively.
- Changed dynamic to static memory allocation for 'array' data member of all k2c_tensor structures to resolve "Unknown size at compile time" error. We declared the array with a size of 725000.

```
#define MAX_TENSOR_ARRAY_SIZE 725000

/**
 * tensor type for keras2c.
 */
struct k2c_tensor
{
    /** Pointer to array of tensor values flattened in row major order. */
    //float * array;
     float array[MAX_TENSOR_ARRAY_SIZE];

    /** Rank of the tensor (number of dimensions). */
    size_t ndim;

    /** Number of elements in the tensor. */
    size_t numel;

    /** Array, size of the tensor in each dimension. */
    size_t shape[K2C_MAX_NDIM];
};
```

- Removed memcpy() functions and replaced them with a 'for' loop.

```
//    memset(C, 0, outrows*outcols*sizeof(C[0]));
    size_t i;

    for (i = 0; i < outrows * outcols; ++i) {
        C[i] = 0;
    }
```

- Removed function pointers by directly calling the function they were pointing to.
- Global declaration of all big arrays and declarations of k2c_tensor structures to solve segmentation fault that was coming due to insufficient stack memory.

❖ **Co-simulation:**

- Changed source file and testbench files' extension from '.c' to '.cpp' files to resolve the below error:

```
INFO: [HLS 200-10] In directory '/home/akshat/original_cnn_struct/original_cnn_struct/solution1/sim/wrapc'
clang: warning: argument unused during compilation: '-fno-builtin-isinf'
clang: warning: argument unused during compilation: '-fno-builtin-isnan'
In file included from /home/akshat/original_cnn_struct/converted.c:1:
In file included from /usr/include/math.h:43:
/usr/include/x86_64-linux-gnu/bits/floatn.h:74:52: error: unsupported machine mode '__TC__'
typedef _Complex float __cfloat128 __attribute__ ((__mode__ (__TC__)));
                                                   ^
/usr/include/x86_64-linux-gnu/bits/floatn.h:86:9: error: unknown type name '__float128'; did you mean '__cfloat128'?
typedef __float128 _Float128;
        ^
/usr/include/x86_64-linux-gnu/bits/floatn.h:74:24: note: '__cfloat128' declared here
typedef _Complex float __cfloat128 __attribute__ ((__mode__ (__TC__)));
                       ^
2 errors generated.
```

3. **Changes made to generate HLS4ML report.**

Commented the below line in ‘build_prj.tcl’ to solve the mentioned error:

Line commented	config_schedule -enable_dsp_full_reg=false
Error	ERROR: [HLS 200-101] 'config_schedule': Unknown option '-enable_dsp_full_reg=false'.:

Following pragmas were removed from ‘nnet_dense_latency.h’ to solve ‘Stop unrolling loop Product-1' error due to pipelining:

#pragma HLS function_instantiate variable=weights,biases
#pragma HLS PIPELINE II=CONFIG_T::reuse_factor
#pragma HLS ARRAY_PARTITION variable=biases complete
#pragma HLS ARRAY_PARTITION variable=mult complete
#pragma HLS ARRAY_PARTITION variable=acc complete
#pragma HLS ALLOCATION operation instances=mul limit=CONFIG_T::multiplier_limit

Following pragmas were removed from pooling2d_cl () function in ‘nnet_pooling.h’ to solve ‘Stop unrolling loop Loop-1' error :

#pragma HLS PIPELINE II=CONFIG_T::reuse_factor
#pragma HLS PIPELINE II=CONFIG_T::reuse_factor
#pragma HLS ARRAY_PARTITION variable=pool complete dim=0

Following pragmas were removed from ‘nnet_activation.h’ to solve ‘Excessive memory usage' error :

#pragma HLS PIPELINE
#pragma HLS UNROLL

#pragma HLS INLINE

Following pragmas in ‘myproject.cpp’ were modified:

Original	Modified
#pragma HLS ARRAY_PARTITION variable=layer2_out complete dim=0	#pragma HLS ARRAY_PARTITION variable=layer2_out block factor=8
#pragma HLS ARRAY_PARTITION variable=layer3_out complete dim=0	#pragma HLS ARRAY_PARTITION variable=layer3_out block factor=8
#pragma HLS ARRAY_PARTITION variable=layer4_out complete dim=0	#pragma HLS ARRAY_PARTITION variable=layer4_out block factor=8
#pragma HLS ARRAY_PARTITION variable=layer6_out complete dim=0	#pragma HLS ARRAY_PARTITION variable=layer6_out block factor=8
#pragma HLS ARRAY_PARTITION variable=layer8_out complete dim=0	#pragma HLS ARRAY_PARTITION variable=layer8_out block factor=8
#pragma HLS ARRAY_PARTITION variable=layer9_out complete dim=0	#pragma HLS ARRAY_PARTITION variable=layer9_out block factor=8
#pragma HLS ARRAY_PARTITION variable=layer10_out complete dim=0	#pragma HLS ARRAY_PARTITION variable=layer10_out block factor=4

These were previously partitioned with type=complete. Due to this, each element of every array was stored in a register which was increasing resource usage and we were getting “Excessive memory usage” error. All arrays except layer10_out was partitioned with type=block and factor=8. The array ‘layer10_out’ was partitioned with type=block and factor=4 as this array’s size is less than 8.

4. **Depedencies and versions required:**

Dependencies

1. Tensorflow version : 2.15.1
2. Pydot
3. Graphviz
4. Python : Python 3.10.x is needed
5. Hls4ml

Commands to install above dependencies

1. pip install tensorflow==2.15.x
2. pip install pydot
3. pip install graphviz
4. sudo apt install python3
5. pip install hls4ml

5. **Optimizations**:

- ❖ Resource optimization:

  The maximum size of ‘array’ data member of k2c_tensor is 720896. Most of the arrays are of size less than 20K. This results in a lot of unused memory.

  For optimizing resource, we removed the k2c_tensor structures completely and wherever we were passing the structure (say) A, we modified that to pass all the 4 parameters of structure : A_array, A_ndim, A_numel, A_shape as pointers.

```
//void k2c_dense_func2(k2c_tensor* output, const k2c_tensor* input, const k2c_tensor2* kernel,
//              const k2c_tensor* bias, int idx, float * fwork) {
void k2c_dense_func2(float* output_array, size_t* output_ndim, size_t* output_numel, size_t* output_shape,
                     const float* input_array, const size_t* input_ndim, const size_t* input_numel, const size_t* input_shape,
                     const float* kernel_array, const size_t* kernel_ndim, const size_t* kernel_numel, const size_t* kernel_shape,
                     const float* bias_array, const size_t* bias_ndim, const size_t* bias_numel, const size_t* bias_shape,
                     int idx, float * fwork) {
```

- ❖ Latency optimization:

For improving latency, we have used array partitioning and loop pipelining. For reference, we have given the snapshot of k2c_affine_matmul() function:

```
// original
void k2c_affine_matmul(float * C, const float * A, const float * B, const float * d,
                       const size_t outrows,const size_t outcols, const size_t innerdim) {

    // make sure output is empty
    size_t i;
     size_t j;
   // memset(C, 0, outrows*outcols*sizeof(C[0]));
     for (int k = 0; k < outrows; ++k) {
             for (int l = 0; l < outcols; ++l) {
                 C[k * outcols + l] = 0.0f;
             }
         }
    for (i= 0 ; i < outrows; ++i) {
        const size_t outrowidx = i*outcols;
        const size_t inneridx = i*innerdim;
        for (j = 0;  j < outcols; ++j) {
            for (size_t k = 0; k < innerdim; ++k) {
                C[outrowidx+j] += A[inneridx+k] * B[k*outcols+j];
            }
            C[outrowidx+j] += d[j];
        }
    }
}

// After optimization
void k2c_affine_matmul(float * C, const float * A, const float * B, const float * d,
                       const size_t outrows,const size_t outcols, const size_t innerdim) {


   #pragma HLS ARRAY_PARTITION variable=B cyclic factor=8

    for (size_t k = 0; k < outrows; ++k) {

        const size_t outrowidx = k * outcols;
        const size_t inneridx = k * innerdim;

        for (size_t l = 0; l < outcols; l+=8) {
               #pragma HLS pipeline II=1

            float sum[8]= {0};
            for(size_t i = 0; i < innerdim; i++)
            {
                float a_val = A[inneridx + i];
                #pragma HLS UNROLL factor = 8
                for(size_t ll=0; ll<8; ll++)
                {
                    sum[ll] += a_val * B[i * outcols + l + ll];
                }
            }

            for(size_t ll=0; ll<8; ll++)
            {
                C[outrowidx + l + ll] = sum[ll] + d[l + ll];
            }
          }
        }
}
```

6. **Results**:

- Latency and area overhead table for Baseline (Unoptimized).

Summary

Name	BRAM_18K	DSP48E	FF	LUT
DSP	-	-	-	-
Expression	-	-	0	626
FIFO	-	-	-	-
Instance	64	167	38730	29763
Memory	30783	-	1056	8
Multiplexer	-	-	-	1900
Register	-	-	1185	-
Total	30847	167	40971	32297
Available	730	740	269200	129000
Utilization (%)	4225	22	15	25

Result

		Latency			Interval		
RTL	Status	min	avg	max	min	avg	max
VHDL	NA	NA	NA	NA	NA	NA	NA
Verilog	Pass	29607917	29607917	29607917	NA	NA	NA

Export the report(.html) using the Export Wizard

- HLS4ML generated Latency and area overhead table.

```
================================================================
== Utilization Estimates
================================================================
* Summary:
+-----------------+---------+-------+--------+--------+
|       Name      | BRAM_18K| DSP48E|   FF   |   LUT  |
+-----------------+---------+-------+--------+--------+
|DSP              |        -|      -|       -|       -|
|Expression       |        -|      -|       0|     270|
|FIFO             |        -|      -|       -|       -|
|Instance         |     1883|    400|  150477|  107674|
|Memory           |       72|      -|    1088|     131|
|Multiplexer      |        -|      -|       -|     531|
|Register         |        -|      -|      59|       -|
+-----------------+---------+-------+--------+--------+
|Total            |     1955|    400|  151624|  108606|
+-----------------+---------+-------+--------+--------+
|Available        |      730|    740|  269200|  129000|
+-----------------+---------+-------+--------+--------+
|Utilization (%)  |      267|     54|      56|      84|
+-----------------+---------+-------+--------+--------+

+ Latency (clock cycles):
    * Summary:
    +---------+---------+---------+---------+----------+
    |      Latency      |      Interval     | Pipeline |
    |   min   |   max   |   min   |   max   |   Type   |
    +---------+---------+---------+---------+----------+
    |  9273017|  9746106|  8685445|  8685445| dataflow |
    +---------+---------+---------+---------+----------+
```

- Latency and area overhead table for resource optimized solution.

Name	BRAM_18K	DSP48E	FF	LUT
DSP	-	-	-	-
Expression	-	-	0	306
FIFO	-	-	-	-
Instance	2086	34	6803	7064
Memory	114	-	0	0
Multiplexer	-	-	-	623
Register	-	-	333	-
Total	2200	34	7136	7993
Available	730	740	269200	129000
Utilization (%)	301	4	2	6

Result

		Latency			Interval		
RTL	Status	min	avg	max	min	avg	max
VHDL	NA	NA	NA	NA	NA	NA	NA
Verilog	Pass	16231732	16231732	16231732	NA	NA	NA

- Latency and area overhead table for latency optimized solution.

Summary

Name	BRAM_18K	DSP48E	FF	LUT
DSP	-	-	-	-
Expression	-	-	0	306
FIFO	-	-	-	-
Instance	2148	139	21364	16557
Memory	116	-	0	0
Multiplexer	-	-	-	1403
Register	-	-	851	-
Total	2264	139	22215	18266
Available	730	740	269200	129000
Utilization (%)	310	18	8	14

Result

		Latency			Interval		
RTL	Status	min	avg	max	min	avg	max
VHDL	NA	NA	NA	NA	NA	NA	NA
Verilog	Pass	4524140	4524140	4524140	NA	NA	NA

Comparison:

Design	LUT	FF	DSP	BRAM	Latency		Clock period(in ns)
					Min	Max	
Baseline	32997	40971	167	30847	29607917	29607917	5
HLS4ML	108606	151624	400	1955	9273017	9746106	5
Resource optimized	7993	7136	34	2200	16231732	16231732	5
Latency optimized	18266	22215	139	2264	4524140	4524140	5