XELERA

XELERA TECHNOLOGIES GMBH, RHEINSTR. 40 – 42, 64283 DARMSTADT

BENCHMARK 001
LENOVO THINKSYSTEM SE350
RANDOM FOREST INFERENCE

V 1.1.0

Benchmark of random forest inference-based recommendation engine on a Lenovo ThinkSystem SE350 edge server utilizing GPU/FPGA accelerator boards

Xelera Technologies GmbH
Alexander Lange
alexander.lange@xelera.io
Additional contributors:
Andrea Suardi
Julian Käuser

Contents

1. Setup

Recently Lenovo has released its ThinkSystem SE350, a server specifically designed for edge cloud environments to move concentrated compute power closer to the end-user. The 1U half-rack is equipped with an Intel Xeon D-2183IT CPU with 16 cores/32 threads @ 2.20 GHz and 128 GB DIMM DDR4 Synchronous 2666 MHz RAM. It is capable of fitting half-length PCIe cards like Xilinx Alveo U50 FPGA (see Figure 1) or NVIDIA Tesla T4 GPU for additional acceleration.

Figure 1 - Lenovo ThinkSystem SE350 with Xilinx Alveo U50

A simple setup has been created to compare the compute capabilities with each accelerator variant and with only the CPU. For all three platforms libraries have been selected that have a very similar API in order to keep the required script-changes close to zero and the comparability as high as possible.

As shown in Figure 2 for the Intel CPU scikit-learn was selected since it provides excellent performance for a lot of machine learning algorithms using CPU-only on single nodes. RAPIDS is a library provided by NVIDIA itself in order to replicate the machine learning capabilities of scikit-learn with improved performance utilizing NVIDIA GPUs. Xelera provides machine learning kernels for FPGAs accordingly as well as a python-interface that is also based on the scikit-learn API.

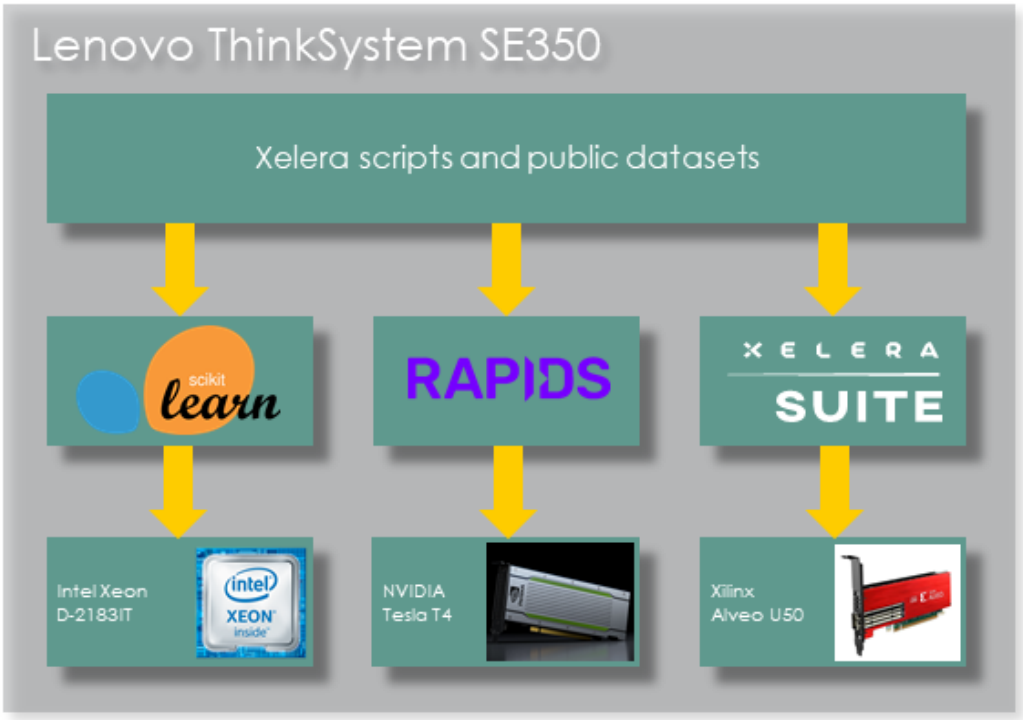

Figure 2 - Benchmark Setup

2. Algorithm

For this benchmark the random forest decision tree inference was selected. The random forest algorithm has a large variety of use cases. One of the most popular ones are so called recommendation engines that can predict future events as used for predictive maintenance to help determine the condition of in-service equipment or than can accurately assess customer needs in order to further personalize services, e.g. by showing more relevant advertisement or recommending fitting media like movies or music. An example of the process is shown in Figure 3.

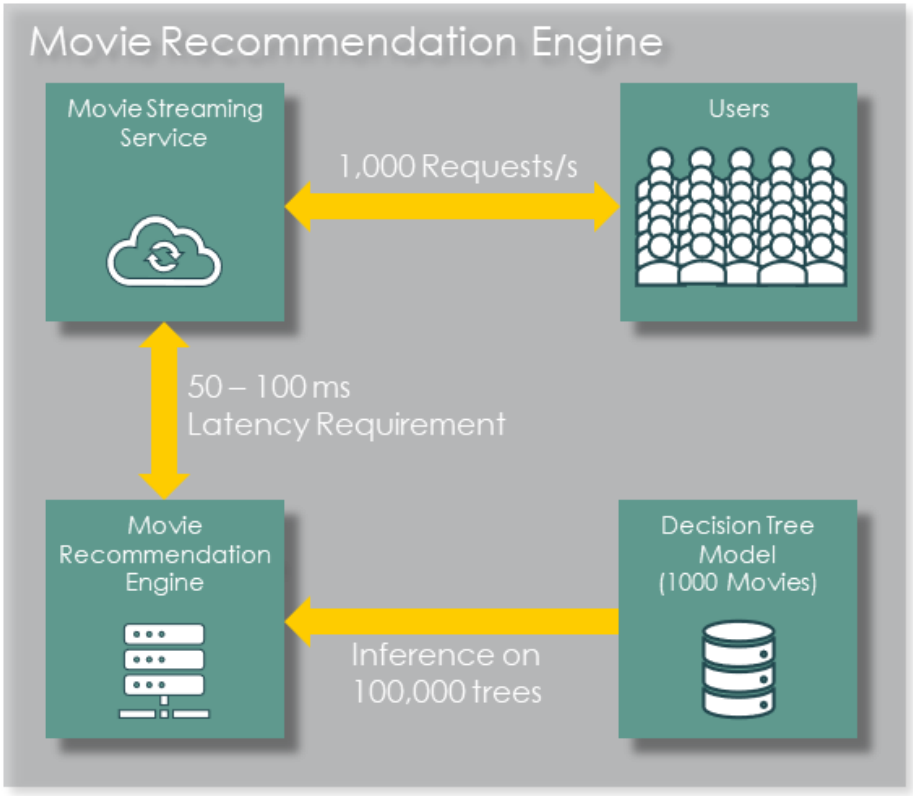

Figure 3 - Typical Recommendation Engine

This typical example shows a movie recommendation engine. This would consist of a huge number of decision trees. Random forests inherently are made out of large numbers of trees, the more the better the accuracy generally. One forest could for example consist of 100 trees. Since one forest would only represent one movie in order to cover a whole database thousands of forests would be needed. Hence the complete model could contain 100,000 trees or more. A large number like this would need to be inferred in real-time in order to conduct the recommendation in a short time window and deliver a good customer experience. A reasonable time window would be to 50 – 100 ms so the information can be delivered to the user without noticeable delay.

3. Benchmark

For the benchmark all three platforms have been tested on the same movie dataset with 16 features. Multiple random forest models of different sizes (number of trees) but each with a fixed depth of 8 levels have been trained. The benchmark was conducted on each platform with each model size as well as a different number of samples (batch size). The respective execution times are shown below.

CPU Execution Time		Trees (Model Size)			
		100	1000	10000	100000
Samples (Batch Size)	1	0.11375	0.55897	5.68215	60.40019
	10	0.11272	0.61511	5.67449	65.80613
	100	0.11358	0.71795	6.46538	65.31643
	1000	0.11094	0.81869	7.68236	77.23859
	10000	0.21596	1.42355	13.90265	136.31746
	100000	1.35437	11.20106	112.67430	1129.38227

Table 1 - Intel Xeon D-2183IT CPU Execution Time [s]

GPU Execution Time		Trees (Model Size)			
		100	1000	10000	100000
Samples (Batch Size)	1	0.01507	0.15316	2.02396	17.36702
	10	0.01560	0.15130	2.07955	17.48893
	100	0.01632	0.15840	2.06146	17.40507
	1000	0.01621	0.15625	2.08708	17.48609
	10000	0.01561	0.15516	2.12261	17.27794
	100000	0.02500	0.17333	2.19849	18.52639

Table 2 - NVIDIA Tesla T4 GPU Execution Time [s]

FPGA Execution Time		Trees (Model Size)			
		100	1000	10000	100000
Samples (Batch Size)	1	0.00054	0.00054	0.00118	0.00413
	10	0.00059	0.00057	0.00116	0.00481
	100	0.00094	0.00095	0.00233	0.01155
	1000	0.00454	0.00464	0.01005	0.05246
	10000	0.04558	0.04416	0.09449	0.43162
	100000	0.37374	0.38121	0.84905	4.17382

Table 3 - Xilinx Alveo U50 FPGA Execution Time [s]

Taking the movie recommendation engine example from chapter 2 we could derive the following parameters:

- 1000 movies → 100,000 trees
- 1000 user requests per second → 1000 samples/s
- 50 – 100 ms real-time constraint

The color coding of the above tables shows how the platforms fair for certain configurations of batch/model sizes:

- green = a single card can fulfill the 50 ms soft requirement
- yellow = a single card can fulfill the 100 ms hard requirement
- red = multiple cards are required to fulfill the 100 ms hard requirement.

A single Xilinx Alveo U50 could easily match the requirements of the example. Two of them could even reduce the latency to a more optimal region. On the other hand, about 1,000 of the Intel CPUs (still slightly above 100 ms) or 163 Nvidia Tesla T4 would be required to achieve the same result.

It gets clear on first sight that the CPU is severely outmatched in several orders of magnitude, especially when bigger models or numbers of samples are used. GPU and FPGA on the other side both have their advantages or disadvantages. To visualize these better the following tables show the speed-up factor between the different platforms. A negative value implies that the platform compared against is faster.

GPU vs. CPU Speed-up		Trees (Model Size)			
		100	1000	10000	100000
Samples (Batch Size)	1	7.55	3.65	2.81	3.48
	10	7.22	4.07	2.73	3.76
	100	6.96	4.53	3.14	3.75
	1000	6.85	5.24	3.68	4.42
	10000	13.84	9.17	6.55	7.89
	100000	54.18	64.62	51.25	60.96

Table 4 - Speed-up NVIDIA T4 vs. Intel Xeon D-2183IT

FPGA vs. CPU Speed-up		Trees (Model Size)			
		100	1000	10000	100000
Samples (Batch Size)	1	211.62	1034.26	4804.66	14625.31
	10	192.19	1074.51	4880.08	13670.81
	100	120.89	755.76	2779.09	5656.03
	1000	24.43	176.47	764.68	1472.21
	10000	4.74	32.24	147.13	315.83
	100000	3.62	29.38	132.71	270.59

Table 5 - Speed-up Xilinx U50 vs. Intel Xeon D-2183IT

FPGA vs. GPU Speed-up		Trees (Model Size)			
		100	1000	10000	100000
Samples (Batch Size)	1	28.04	283.39	1711.40	4205.25
	10	26.61	264.29	1788.41	3633.22
	100	17.37	166.74	886.10	1507.18
	1000	3.57	33.68	207.74	333.29
	10000	-2.92	3.51	22.46	40.03
	100000	-14.95	-2.20	2.59	4.44

Table 6 - Speed-up Xilinx U50 vs. NVIDIA T4

4. Conclusion

As Table 4 and Table 5 show both accelerators have a speed advantage over CPUs in all test cases. The biggest improvement can be seen when big batch sizes (number of samples) are used for GPUs or when big models (number of trees) are used for FPGAs.

Table 6 makes clear that FPGAs have a significant edge over GPUs as well in most cases, especially for big model sizes. For big batch sizes and small models the GPU is a bit faster. It is obvious that both accelerators have their area where they perform best and it depends on the use-case which one should be selected.

Since the use-case of this report is to achieve real-time utilizing the edge cloud the FPGA has clearly the upper hand over its two competitors. When compute power is moved from the cloud closer to the edge the number of samples that need to be inferred within a certain time window inherently gets smaller. On the other hand, random forest models are naturally very large, as shown in the example calculation of chapter 3. In both areas the FPGA showed excellent performance. It is the accelerator of choice to achieve true real-time.

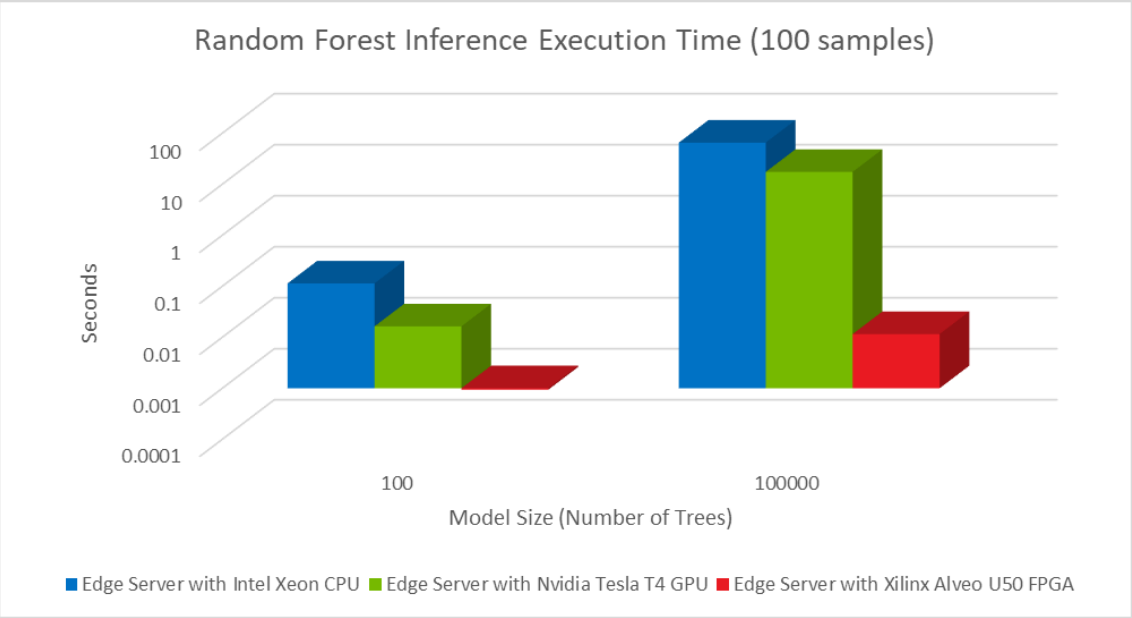

Figure 4 - Random Forest Inference Execution Time (100 samples)

Annex A: Required devices by minimum samples per second

Assuming each sample (request) has to be processed within 100 ms latency the following tables show how many devices of each type would be required to process a certain number of samples per second for a specific model size.

CPU Required Devices		Trees (Model Size)			
		100	1000	10000	100000
Samples/s	1000	2	8	65	654
	10000	2	9	77	773
	100000	3	15	140	1364
	1000000	14	113	1127	11294

Table 7 - Intel Xeon D-2183IT CPU Required Devices (100 ms max latency)

GPU Required Devices		Trees (Model Size)			
		100	1000	10000	100000
Samples/s	1000	1	2	21	175
	10000	1	2	22	175
	100000	1	2	22	175
	1000000	1	2	22	186

Table 8 - NVIDIA Tesla T4 GPU Required Devices (100 ms max latency)

FPGA Required Devices		Trees (Model Size)			
		100	1000	10000	100000
Samples/s	1000	1	1	1	1
	10000	1	1	1	1
	100000	1	1	1	5
	1000000	4	4	9	42

Table 9 - Xilinx Alveo U50 FPGA Required Devices (100 ms max latency)