Stairway to AI: Ease the Engagement of Low-Tech users to the AI-on-Demand platform through AI, H2020

Benchmarking results-1st

Deliverable information	
Deliverable number	D6.3
WP number and title	WP6 - Vertical Matchmaking and hardware marketplace
Lead beneficiary	BCA
Dissemination level	Public
Due date	31 January 2023
Actual date of delivery	13 February 2023
Author(s)	Miguel de Prado (BCA), Andrea Borghesi (Unibo), Jacopo Carletti (INFN)

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 101017142

Document Control Sheet

Version	Date	Summary of changes	Author(s)
0.1	11/01/2023	Initial draft of document	Miguel de Prado (BCA)
0.3	26/01/2023	First draft complete	Miguel de Prado (BCA), Andrea Borghesi (Unibo), Jacopo Carletti (INFN)
0.7	10/02/2023	Corrections after QA review	Miguel de Prado (BCA)
1.0	13/02/2023	Final document	Miguel de Prado (BCA)

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017142

Table of contents

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017142

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017142

Acronyms

Acronym	Explanation
AI	Artificial Intelligence
API	Application Programming Interface
BMP	Bonseyes Marketplace
DNN	Deep Neural Network
LPDNN	Low-power Deep Neural Network framework
ML	Machine Learning
WP	Work Package
CPU	Central Processing Unit
GPU	Graphical Processing Unit
NPU	Neural Processing Unit

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017142

1. Executive Summary

This document delivers the first version of the *Benchmark results* proposed within WP6 Task 6.2 of StairwAI project. The ***benchmark results*** **are the outcome of** ***benchmark software framework*** executed across multiple heterogenous platforms, which are then used **to train the vertical matchmaking engine** (Task 6.3).

Other than the results obtained from the *benchmark software framework (Section 2)*, this document also presents a summary of external benchmarks (Section 3) that are commonly used by researchers to push the limits of the fields. The combination of the external benchmarks with the ones provided by the *benchmark software framework* allows us to have a richer and more complete database to train the vertical matchmaking engine, which is discussed in Section 4. Lastly, Section 5 elaborates the conclusions and future work on Task 6.2.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017142

2. Introduction

As the computational capacity of embedded devices grows, they become more capable of performing sophisticated tasks, including those considered artificial intelligence (AI) applications. Since the advent of AI for computer vision and speech recognition tasks, training of AI models has been performed using machine learning algorithms, with a large number of open-source datasets created by researchers. Such datasets provide an opportunity to develop, train and test AI algorithms on devices that range from the Cloud to the Edge.

A significant concern for AI engineers during the deployment of AI Algorithms is maximizing the system's performance in terms of overall latency, quality of solution, power, and space. The evaluation of AI Algorithms is often based on how well they perform on restricted public benchmark problems. These benchmarks give an informative indication of how well a given algorithm will perform on some given tested systems. However, AI Algorithms are often deployed on systems with rather different characteristics. As a result, the performance can vary widely according to the system on which they are deployed. Therefore, it is vital to have a large variety of benchmarks across multiple heterogeneous platforms to understand an AI Algorithm's behaviour comprehensively.

Taking these principles, the main contributions of this deliverable are the following:

- Presentation of benchmark results obtained by StairwAI's benchmark frameworks across heterogeneous platforms, including CPUs, GPUs, and NPUs.
- Survey of external benchmarks, presenting their scope, metrics, pros, and cons.
- Discussion about the found results towards creating a Vertical Matchmaking Engine.

2.1. Purpose and Scope of the document

WP6 has the main objective of building a Vertical Matchmaking engine that matches AI algorithms and HW resources to optimise the deployment of services and increase their efficiency. In the deliverable D6.1 (Benchmarking software-1st version M20), the *benchmark software framework* was introduced, which had the main objectives of developing a software tool to benchmark machine learning models and hardware, including CPU (Intel x86, Arm Cortex-A5x, Risc-V), GPGPU (NVIDIA, etc), and NPU (HUA, etc) accelerated platforms.
Deliverable D6.3 is a Report, i.e., it introduces the first version of the *Benchmark results,* which will be extended during the course of Task 6.2 and will conclude with D6.4 (Benchmarking results-2nd M30). In this deliverable, we present the benchmarking data set collected so far by the *benchmark software framework*. Further, we offer the results of a survey we conducted to investigate existing benchmarking data, especially to understand whether it could be useful for the Vertical Matchmaking Engine (D6.5 in M30). Finally, we summarize the key insights gathered and provide some observations. We will explicitly focus the discussion on the aspects most relevant for the downstream task in WP6, namely the vertical matchmaking engine – which will exploit the benchmarking data to reach its goals. In addition, we also provide some key insights about the alignment with external benchmark frameworks and data from the AI-on-demand platform.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017142

3. Benchmark results (by StairwAI)

3.1. Benchmark Software Framework (LPDNN)

3.1.1. Scope

LPDNN is the deployment framework that was introduced in D6.1 as the *benchmark software framework* developed during T6.1. The LPDNN framework was initially developed during the H2020 Bonseyes project (Prado, Miguel De, et al) and has been largely extended in StarwAI to provide a structured benchmarking layer to analyze the execution of AI Applications on a variety of heterogeneous platforms. This layer, on top of LPDNN, adheres to the following design principles:

- ❑ Industry-driven Research:
  - ➢ Input requirements from SMEs wanting to use AI solutions.
  - ➢ Able to deploy AI solutions on edge devices (low-power, low-carbon footprint).
- ❑ Structured benchmarking workflow:
  - ➢ Availability of anchors within the deployment framework to evaluate the metrics truthfully.
  - ➢ Optimised deployment (value added to research and industry).
  - ➢ Extensive documentation & Support (user friendly for SMEs).
  - ➢ Defined interfaces (standarization).
  - ➢ Easy to replicate (reproducibility).
  - ➢ Create trust & community.

LPDNN provides the tools and capabilities to generate portable and efficient AI applications, which can be deployed and optimised across heterogeneous platforms, e.g., CPU, GPU, FPGA, NPU (ASIC). LPDNN features a full development flow for AI solutions on hardware devices by providing platform support, sample models, optimisation tools, integration of external libraries, and benchmarking.

LPDNN's full development flow makes the AI solution reliable and easy to replicate, increasing the **portability and fairness across the wide span of hardware platforms**. LPDNN's flexible architecture allows the main core to remain small and dependency-free. At the same time, additional 3rd party libraries or inference engines are only included when needed and for specific platforms, notably increasing the portability across systems. Thereby, different systems can be benchmarked with the same benchmarking code and fairly compared.

The collected dataset using LPDNN across heterogenous platforms has been open sourced at: https://gitlab.com/bonseyes/bonseyes-benchmarks/-/tree/master.

Almost 3000 implementations have been benchmarked across 8 different HW platforms, allowing the first version of the Vertical Matchmaking engine to be trained (T6.3, to be reported in D6.5 in M30).

For more details about LPDNN, please refer to D6.1.

3.1.2. Metrics

LPDNN provides a first analysis of static metrics during the offline compilation of the AI Applications. The offline metrics are the following:

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017142

- **AI_TASK**: Task or class that the AI Model does, e.g., image classification.
- **MODEL_VERSION**: Specific version of the AI Model, e.g., different backbone.
- **INPUT_TILE**: Input size that the neural network takes.
- **PLATFORM**: HW platform where the AI Model is deployed.
- **ENGINE**: Inference engine that executes the AI Model.
- **PROCESSOR**: Processor where the AI Model is executed.
- **PRECISION**: Precision, e.g., floating point, of the data.
- **GFLOP**: Giga Floating Point OPerations that the AI Model's inference contains
- **#PARAMS**: Number of parameters (weights) that the AI Model contains.
- **STORAGE**: Size on disk of #PARAMS.

During the execution of the AI Applications, LPDNN measures the following metrics:

- **QUALITY_METRIC**: Quality metric, e.g., accuracy, mean square error.
- **QUALITY_VALUE**: Value of the quality metric.
- **PREPROCESSING_TIME**: Time spent on pre-processing the data.
- **INFERENCE_TIME**: Time spent on the forward pass of the neural network (NN).
- **POSTPROCESSING_TIME**: Time spent on post-processing the results of the NN.
- **LATENCY**: Addition of pre/post-processing+ inference times.
- **THROUGHPUT**: Number of executions per second. Inverse of latency.
- **DMIPS**: Dhrystone Million Instructions per Second.
- **CPU_MEM**: Average CPU memory allocated over the execution.
- **CPU_MEM_PEAK**: Peak CPU memory allocated over the execution.
- **GPU_MEM**: Average CPU memory allocated over the execution.
- **GPU_MEM_PEAK**: Peak CPU memory allocated over the execution.
- **MEMORY_BANDWIDTH**: Average Memory bandwidth taken.
- **MEMORY_BANDWIDTH**_PEAK: Peak Memory bandwidth taken.
- **CPU_LOAD**: Average load of the CPU over the execution.
- **GPU_LOAD**: Average load of the GPU over the execution.
- **NPU_LOAD**: Average load of the NPU over the execution.
- **CPU_TEMP**: Average temperature of the CPU over the execution.
- **GPU_TEMP**: Average temperature of the GPU over the execution.
- **POWER_CONSUMPTION**: Power consumption over the execution.
- **ENERGY_EFFICIENCY**: Energy efficiency over the execution.

For more detail about how these metrics are collected, please refer to D6.1.

Among these metrics, it is worth noting the different supported platforms, inference engines and AI models:

Platforms:

- **Raspberry Pi 3b+:** Quad-core ARM Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz
- **Raspberry Pi 4b:** Quad-core ARM Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
- **Nvidia Jetson Nano**: Quad-core ARM Cortex -A57 @ 1.43 GHz & 128-core Nvidia Maxwell GPU
- **Nvidia Jetson Xavier:** Octa-core ARM v8.2 @ 2.03 GHz & 512-core Nvidia Volta GPU with Tensor Cores

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017142

- **Intel NUC:** Intel quad-core (TM) i5-7260U CPU @ 3.4 GHz
- **iMX8m Nano:** Quad-core ARM Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz
- **STM32 MP1:** Dual-core ARM Cortex-A7 cores up to @ 800 MHz
- **HPC**: General x86 Intel or AMD CPU cores and Nvidia GPUs

Inference engines:

- **LNE**: LPDNN Native Engine (LNE) allows the execution of DNNs across arm-based and x86 CPUs as well as on Nvidia-based GPUs.
- **NCNN**: NCNN ports the execution of DNNs on GPU through the Vulkan driver.
- **TensorRT**: TensorRT accelerates the DNN inference on Nvidia-based GPUs and NPUs.
- **ONNXruntime**: ONNXruntime enables the direct execution of ONNX models on CPUs and GPUs.

AI Models:

- face-landmarks-detection-3ddfa
- body-pose-openpifpaf
- face-landmarks-detection-retinaface
- face-recognition-insightface-mobilefacenet
- incar-object-detection-nanodet
- age-gender-insightface
- emotion-classic-light-112x96
- eyegaze-rankgaze
- headpose-rankpose
- imagenet-alexnet
- imagenet-googlenet
- imagenet-efficientb0
- imagenet-squeezenet
- imagenet-mobilenet
- imagenet-mobilenet-v2
- imagenet-mobilenet-v3
- imagenet-resnet-resnet18
- imagenet-resnet-resnet32
- imagenet-resnet-resnet50
- coco-nanodet

3.1.3. Pros/Cons

PROS

- **Fair comparison across SW/HW vendors:**
  - Integration of SW inference engines from several vendors on the same codebase
  - Execution on heterogeneous systems (CPU, GPU, NPU)
- **Variety on Model deployment:**
  - Multiple flavours of models and backbones with several input sizes
  - Deployment with different inference engines and data types (fp32, fp16, int8)

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017142

- **Real Industrial benchmark**
  - Measures real latency of AI App with pre- & post-processing s
  - Monitors system during benchmarking
  - Extensive number of metrics

CONS

- Mostly focused on deep learning model for computer vision
- Relatively small amount of data

3.2. Benchmark online algorithms (Energy domain)

3.2.1. Scope

Online algorithms configuration for energy systems is a complex domain area characterized by uncertainty, e.g., renewable energy production, demand fluctuation, tight Hardware (HW) and real-time constraints. An online algorithm should be able to calculate the amount of energy that must be produced by the energy system to meet the required load, minimizing the total energy cost over the daily time horizon and by taking into account the uncertainty. This is a typical real-world problem with real time constraints.

Selecting the optimal HW configuration for a given set of tight constraints on solution time and quality over multiple and diverse data instances (e.g., the specific details of the energy system) for a given online algorithm is a complex problem. Moreover, each algorithm is characterized by a configurable parameter that further complicates this task.

The collected dataset using online algorithms has been open sourced at
https://zenodo.org/record/5838437

For more details about the dataset, please refer to https://doi.org/10.1016/j.knosys.2022.109199

3.2.2. Metrics

The benchmark describes the behaviour of two different online algorithms with different parameter configurations and with different input instances (e.g., the specific details of the energy system and renewable energy production and demand fluctuation). In more details, both algorithms are characterized by a single changeable parameter that impacts the measured metrics: algorithm runtime, memory consumption and solution quality.

For each record of the dataset, we collected the following information:

- **nParameter**: an integer that represents the number of scenario/traces used by the algorithm (the configurable parameter).
- **Load (kW)**: a vector of 96 values representing the load observations sampled in 96 stages (every 15 minutes over the course of a day).
- **RES (kW)**: a vector of 96 values representing the observations of available renewable production.

During the execution of the different algorithm configurations, we measure the following metrics:

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017142

- **solution quality (k\euro)**: a real number representing the value of the solution, in practice a measure of the daily total energy cost. It is obtained as the sum over the entire time horizon (over 96 stages) of all partial solutions.
- **runtime (sec)**: the time required for finding the solution with the algorithm. It is obtained as sum over the entire time horizon of all 96 partial running times.
- **memory consumption (MB)**: represents the RAM used by the algorithm on the machine where it was performed. It is an average of the memory used by the algorithm in 96 partial runs.

The dataset is generated by executing both the algorithms on different problem instances and with different parameter configurations.

Platform:

All the runs for dataset creation have been performed on Intel Core i5 (3,1 GHz) machines with 16 GB of RAM.

3.2.3. Pros/Cons

CONS

The limitation of this dataset is mainly that we focused on a single HW architecture for analyzing the performance of both the algorithms with different parameter's configurations.

PROS

This dataset represents the first prototype to test the vertical MatchMaking (MM) engine:

- exhaustive grid-search exploration of the hyperparameter space
- exhaustive validation of the vertical MM engine on different configurations.

4. Survey of external benchmarks

4.1. AI Benchmark

4.1.1. Scope

The AI Benchmark is an Android application designed to check the performance and the memory limitations associated with running AI and deep learning algorithms on mobile platforms. It consists of several computer vision tasks performed by neural networks that are running directly on Android devices. The considered networks represent the most popular and commonly used architectures that can be currently deployed on smartphones.

Besides the Android version, a separate open-source AI Benchmark build for desktops was released in June 2019. It is targeted at evaluating AI performance of the common hardware platforms, including CPUs, GPUs and TPUs, and measures the inference and training speed for several key deep learning models. The benchmark is relying on the TensorFlow machine learning library and is distributed as a Python pip package that can be installed on any system running Windows, Linux or macOS.

Reference Paper: https://arxiv.org/pdf/1810.01109.pdf, https://arxiv.org/pdf/1910.06663.pdf

Website: https://ai-benchmark.com/index.html

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017142

4.1.2. Metrics

Supported Mobile Architectures:

- Qualcomm: Snapdragon 845 (Hex. 685 + Adreno 630); Snapdragon 710 (Hexagon 685); Snapdragon 670 (Hexagon 685); Snapdragon 855+ (Hex. 690 + Adreno 640); Snapdragon 855 (Hex. 690 + Adreno 640); Snapdragon 730 (Hex. 688 + Adreno 618); Snapdragon 675 (Hex. 685 + Adreno 612); Snapdragon 665 (Hex. 686 + Adreno 610).
- HiSilicon: Kirin 970 (NPU, Cambricon); Kirin 980 (NPU×2, Cambricon).
- Samsung: Exynos 9810 (Mali-G72 MP18); Exynos 9610 (Mali-G72 MP3); Exynos 9609 (Mali-G72 MP3); Exynos 9825 (NPU + Mali-G76 MP12); Exynos 9820 (NPU + Mali-G76 MP12).
- MediaTek: Helio P70 (APU 1.0 + Mali-G72 MP3); Helio P60 (APU 1.0 + Mali-G72 MP3); Helio P65 (Mali-G52 MP2; Helio P90 (APU 2.0); Helio G90 (APU 1.0 + Mali-G76 MP4).

Frameworks:

- TensorFlow Mobile
- TensorFlow Lite
- Caffe2

Deep Learning Tests:

- **Test Section 1: Image Classification**, Model: MobileNet-V2, Inference modes: CPU (FP16/32) and NNAPI (INT8 + FP16), Image resolution: 224×224 px, Test time limit: 20 seconds.
- **Test Section 2: Image Classification**, Model: Inception-V3, Inference modes: CPU (FP16/32) and NNAPI (INT8 + FP16), Image resolution: 346X346 px, Test time limit: 30 seconds.
- **Test Section 3: Face Recognition**, Model: Inception-ResNet-V1, Inference modes: CPU (INT8) and NNAPI (INT8 + FP16), Image resolution: 512X512 px, Test time limit: 30 seconds.
- **Test Section 4: Playing Atari**, Model: LSTM, Inference modes: CPU (FP16/32), Image resolution: 84X84 px, Test time limit: 20 seconds.
- **Test Section 5: Image Deblurring**, Model: SRCNN 9-5-5, Inference modes: NNAPI (INT8 + FP16), Image resolution: 384X384 px, Test time limit: 30 seconds.
- **Test Section 6: Image Super-Resolution**, Model: VGG-19 (VDSR), Inference modes: NNAPI (INT8 + FP16), Image resolution: 256X256 px, Test time limit: 30 seconds.
- **Test Section 7: Image Super-Resolution**, Model: SRGAN, Inference modes: CPU (INT8 + FP16/32), Image resolution: 512X512 px, Test time limit: 40 seconds.
- **Test Section 8: Bokeh Simulation**, Model: U-Net, Inference modes: CPU (FP16/32), Image resolution: 128X128 px, Test time limit: 20 seconds.
- **Test Section 9: Image Segmentation**, Model: ICNet, Inference modes: NNAPI (2 X FP32 models in parallel), Image resolution: 768X1152 px, Test time limit: 20 seconds.
- **Test Section 10: Image Enhancement**, Model: DPED-ResNet, Inference modes: NNAPI (FP16 + FP32), Image resolution: 128X192 px, Test time limit: 20 seconds.
- **Test Section 11: Memory Test**, Model: SRCNN 9-5-5, Inference modes: NNAPI (FP16), Image resolution: from 200X200 px to 2000X2000 px.

Desktop GPU and CPUs architectures:

- ➢ Tesla V100 SXM2 32Gb

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017142

- Tesla V100 PCIE 32Gb
- NVIDIA Quadro GV100
- NVIDIA Quadro RTX 8000
- GeForce RTX 2070 SUPER
- NVIDIA TITAN Xp CE
- AMD Radeon VII
- GeForce RTX 2080 Max-Q
- GeForce RTX 2060 Laptop
- NVIDIA Tesla T4
- Intel Xeon Gold 6148
- Intel Xeon Gold 6248
- AMD EPYC 7451

Desktop GPU and CPUs datasets and tasks

- MobileNet-V2 [classification]
- Inception-V3 [classification]
- Inception-V4 [classification]
- Inception-ResNet-V2 [classification]
- ResNet-V2-50 [classification]
- ResNet-V2-152 [classification]
- VGG-16 [classification]
- SRCNN9-5-5 [image-to-image mapping]
- VGG-19 [image-to-image mapping]
- ResNet-SRGAN [image-to-image mapping]
- ResNet-DPED [image-to-image mapping]
- U-Net [image-to-image mapping]
- Nvidia-SPADE [image-to-image mapping]
- ICNet [image segmentation]

Scoring System

AI Benchmark is measuring the performance of several test categories, including int-8, float-16, float-32, parallel, CPU (int-8 and float-16/32), memory tests, and tests measuring model initialization time.

The contribution of the test categories is as follows:

- 48%-float-16 tests;
- 24%-int-8 tests;
- 12%-CPU,float-16/32 tests;
- 6%-CPU,int-8 tests;
- 4%-float-32 tests;
- 3%-parallel execution of the models;
- 2%-initialization time, float models;
- 1%-initialization time, quantized models;

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017142

The scores of each category are computed as a geometric mean of the test results belonging to this category. The computed L1 error is used to penalize the runtime of the corresponding networks running with NNAPI (an exponential penalty with exponent 1.5 is applied).

4.1.3. Pros/Cons

PROS

- **Heavy focus on computer vision**
- **Multiple ML architecture**:
  - MobileNet, Inception
- **Multiple precision for the same ML architecture**:
  - FP16, FP8
- **State of Art Platforms/hardware**:
  - AMD processors, Nvidia accelerators, etc
- **Industry standard and unbiased evaluations**:
  - Recognised by major companies and universities.
- **Regularly updated with new platforms**

CONS

- **Mobile Oriented:** CPU and GPU results are not easy to compare with Mobile Inference. Still under development
- **Relies on TensorFlow Lite:** the number of critical bugs and issues introduced in its new versions prevents from recommending it for any commercial projects or projects dealing with non-standard AI models

4.2. MLPerf

4.2.1. Scope

MLPerf aims to create a representative benchmark suite for ML that evaluates system performance to meet five high-level goals:

1. Enable fair comparison of competing systems while still encouraging ML innovation.
2. Accelerate ML progress through fair and useful measurement.
3. Enforce reproducibility to ensure reliable results.
4. Serve both the commercial and research communities.
5. Keep benchmarking effort affordable so all can participate.

Reference Paper: https://arxiv.org/abs/1911.02549

Website: https://mlcommons.org/en/, section “Benchmarks”, subsections “Training” and “Inference”

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 101017142

Training

MLPerf Training does the following:

1. Establish a comprehensive benchmark suite that covers diverse applications, DNN models, and optimizers.
2. Create reference implementations of each benchmark to precisely define models and training procedures.
3. Establish rules that ensure submissions are equivalent to these reference implementations and use equivalent hyperparameters.
4. Establish timing rules to minimize the effects of stochasticity when comparing results.
5. Make submission code open source so that the ML and systems communities can study and replicate the results.
6. Form working groups to keep the benchmark suite up to date.

ML areas, including vision, language, recommendation, and reinforcement learning set of seven benchmarks.

Inference

ML inference systems range from deeply embedded devices to smartphones to data centers. They have a variety of real-world applications and many figures of merit, each requiring multiple performance metrics. The right metrics, reflecting production use cases, allow not just MLPerf but also publications to show how a practical ML system would perform. MLPerf Inference consists of four evaluation scenarios: single-stream, multistream, server, and offline.

These scenarios represent many critical inference applications. MLPerf Inference provides a way to simulate the realistic behavior of the inference system under test.

4.2.2. Metrics

Training

Area	Benchmark	Dataset	Quality Threshold	Model
Vision	Image classification	ImageNet	75.90% classification	ResNet-50 v1.5
Vision	Image segmentation (medical)	KiTS19	0.908 Mean DICE score	3D U-Net
Vision	Object detection (light weight)	Open Images	34.0% mAP	RetinaNet
Vision	Object detection (heavy weight)	COCO	0.377 Box min AP and 0.339 Mask min AP	Mask R-CNN
Language	Speech recognition	LibriSpeech	0.058 Word Error Rate	RNN-T

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017142

Language	NLP	Wikipedia 2020/01/01	0.72 Mask-LM accuracy	BERT-large
Commerce	Recommendation	1TB Click Logs	0.8025 AUC	DLRM
Research	Reinforcement learning	Go	50% win rate vs. checkpoint	Mini Go (based on Alpha Go paper)

Training Supported Hardware

- NC96ads_A100_v4
- ND96amsr_A100_v4_n16
- ND96amsr_A100_v4_n8
- ESCN4A-E11
- ESC8000A-E11-8xA100-PCIE-80GB-NVBridge
- 8_node_64_A100_PaddlePaddle
- R750xax4A100-PCIE-80GB
- PRIMERGY-RX2540M6-mxnet
- 8xR750xax4A100-PCIE-80GB
- G492-ZD2
- HPE-ProLiant-XL675d-Gen10-Plus_A100-SXM-80GB_hugectr
- HLS-Gaudi2-PT
- Dell Precision 7920 Tower with 2x A5000 using MxNet 22.04
- Lenovo ThinkSystem SR670 V2 Server with 4x 40GB SXM4 A100
- Lenovo ThinkSystem SR670 V2 Server with 8x 80GB PCIe A100
- dgxa100_ngc22.04_merlin_hugectr
- AS-4124GS-TNR
- G5500V6x8xA30
- 1-node-SPR-pytorch
- 16-nodes-SPR-pytorch
- dgxh100_n4_preview
- NF5468M6J

Training HPC

Area	Benchmark	Dataset	Quality Threshold	Model
Scientific	Climate segmentation	CAM5+TECA simulation	IOU 0.82	DeepCAM
Scientific	Cosmology parameter prediction	CosmoFlow N-body simulation	Mean average error 0.124	CosmoFlow
Scientific	Quantum molecular modeling	Open Catalyst 2020 (OC20)	Forces mean absolute error 0.036	DimeNet++

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017142

Time-to-Train Performance Metric

To address the ML-benchmarking challenges of system optimization and scale, MLPerf performance metric is the time to train to a defined quality target.

It incorporates both system speed and accuracy and is most relevant to ML practitioners. As an end-to end metric, it also captures the auxiliary operations necessary for training such models, including data-pipeline and accuracy calculations. The metric's generality enables application to reinforcement learning, unsupervised learning, generative adversarial networks, and other training schemes.

Each benchmark measures the wall clock time required to train a model on the specified dataset to achieve the specified quality target. To account for the substantial variance in ML training times, final results are obtained by measuring the benchmark a benchmark-specific number of times, discarding the lowest and highest results, and averaging the remaining results. Even the multiple result average is not sufficient to eliminate all variance. Imaging benchmark results are very roughly +/- 2.5% and other benchmarks are very roughly +/- 5%.

For non-HPC training, results that converged in fewer epochs than the reference implementation run with the same hyperparameters were normalized to the expected number of epochs.

Inference

MLPerf defines model-quality targets. We established per-model and scenario targets for inference latency and model quality. The latency bounds and target qualities are based on input gathered from ML-system end users and ML practitioners. As MLPerf improves these parameters in accordance with industry needs, the broader research community can track them to stay relevant.

Inference Datacenter

Area	Benchmark	Dataset	Quality Threshold	Model
Vision	Image classification	ImageNet (224x224)	99% of FP32 (76.46%)	Resnet50-v1.5
Vision	Object detection	OpenImages (800x800)	99% of FP32 (0.20 mAP)	Retinanet
Vision	Medical image segmentation	KITS 2019 (602x512x512)	99% of FP32 and 99.9% of FP32 (0.86330 mean DICE score)	3D UNET
Speech	Speech-to-text	Librispeech dev-clean (samples < 15 seconds)	99% of FP32 (1 - WER, where WER=7.452253714852645%)	RNNT
Language	Language processing	SQuAD v1.1 (max_seq_len=384)	99% of FP32 and 99.9% of FP32 (f1_score=90.874%)	BERT-large
Commerce	Recommendation	1TB Click Logs	99% of FP32 and 99.9% of FP32 (AUC=80.25%)	DLRM

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017142

Metrics:

A. queries/s
B. samples/s
C. Accuracy
D. System Power (W)

Inference Mobile

Area	Benchmark	Dataset	Quality Threshold	Model
Vision	Image classification	ImageNet	98% of FP32 (Top1: 76.19%)	MobileNetEdgeTPU
Vision	Object detection	MS-COCO 2017	95% of FP32 (mAp: 0.285)	MobileDETs
Vision	Segmentation	ADE20K (32 classes, 512x512)	97% of FP32 (32-class mIOU: 54.8)	DeepLabV3+ (MobileNetV2)
Vision	Segmentation, MOSAIC	ADE20K (32 classes, 512x512)	96% of FP32 (32-class mIOU: 59.8)	MOSAIC
Language	Language processing	SQUAD 1.1	93% of FP32 (F1 score: 90.5)	Mobile-BERT

Metrics:

A. frames/s
B. latency in ms

4.2.3. Pros/Cons

PROS

The framework offers different sorts of benchmarks for comparison/innovation:

- **Close:** Same original model on HW or SW framework
- **Open:** - Allow to innovate with networks and frameworks
- **Power:** - Allows to obtain system power measurements
- **Scenarios:** - Different streams: single, multi, server, offline
- **Platforms:** - Many HW platforms and SW frameworks

CONS

- Only few tasks
- Only few models
- No memory usage
- No system usage
- No pre- or post-processing

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017142

4.3. AIBench

4.3.1. Scope

AIBench provides a scalable and comprehensive data-center AI benchmark suite. In total, it includes 12 micro benchmarks, 16 component benchmarks, covering 16 AI problem domains: image classification, image generation, text-to-text translation, image-to-text, image-to-image, speech-to-text, face embedding, 3D face recognition, object detection, video prediction, image compression, recommendation, 3D object reconstruction, text summarization, spatial transformer, learning to rank, and two end-to-end application. AI benchmarks:

- DCMix —a datacenter AI application combination mixed with AI workloads
- E-commerce AI—an end-to-end business AI benchmark.

Reference Paper: https://arxiv.org/pdf/2004.14690, http://www.benchcouncil.org/aibench/file/AIBench-Bench18.pdf, https://www.benchcouncil.org/file/Cluster_2021_hpcai_camera.pdf

Website: https://www.benchcouncil.org/aibench/index.html, section "AIBench Training" and "AIBench Inference"

4.3.2. Metrics

AI Tasks

- **Image generation** uses WGAN algorithms and uses LSUN dataset as data input to generate image data.
- **Text-to-Text Translation** uses recurrent neural networks and takes WMTEnglish-German as data input to translate text data.
- **Image-to-Text** uses Neural Image Caption model and takes Microsoft COCO dataset as input to describe image using text.
- **Image-to-Image** uses the cycleGAN algorithm and takes Cityscapes dataset as input to transform the image to another image.
- **Speech-to-Text** uses the DeepSpeech2 algorithm and takes Librispeech dataset as input to recognize the speech data.
- **Face embedding** uses the FaceNet algorithm and takes the LFW (Labeled Faces in the Wild) dataset or VGGFace2 as input to convert image to an embedding vector.
- **3Dface recognition** uses 3D face modes to recognize 3D information within images. The input data includes 77,715 samples from 253 face IDs, which is published on the BenchCouncil web site.
- **Object detection** uses the Faster R-CNN algorithm and takes Microsoft COCO dataset as input to detect objects in images.
- **Recommendation** uses collaborative filtering algorithm and takes MovieLens dataset as input to provide recommendations.
- **Video prediction** uses motion-focused predictive models and takes Robot pushing dataset as input to predict video frames.
- **Image compression** uses recurrent neural networks and takes ImageNet dataset as input to compression images.
- **3D object reconstruction** uses a convolutional encoder-decoder network and takes ShapeNet Dataset as input to reconstruct 3D object.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017142

- **Text summarization** uses sequence-to-sequence model and takes Gigaword dataset as input to generate summary description for text.
- **Spatial transformer** uses spatial transformer networks and takes MNIST dataset as input to make spatial transformations.
- **Learning to Rank** uses ranking distillation algorithm and uses Gowalla dataset to generate ranking scores.

Metrics

- Wall Clock Time
- Energy consumption of running a benchmark
- Accuracy

Provides both training and inference benchmarks.

The training metrics are the wall clock time to train the specific epochs, the wall clock time to train a model achieving a target accuracy, and the energy consumption to train a model achieving a target accuracy.

The inference metrics are the wall clock time, accuracy, and energy consumption.

Additionally, the performance numbers are reported on the BenchCouncil web site (http://www.benchcouncil.org/numbers.html), to measure the training and inference speeds of different hardware platforms, including multiple types of NIVDIA GPUs, Intel CPUs, AI accelerator chips, and to measure the performance of different software stacks, including TensorFlow, PyTorch, and etc.

4.3.3. Pros/Cons

PROS

- **Wide variety of data types** and data sources are covered, including text, images, street scenes, audios, videos, etc.
- **Not only based on mainstream deep learning frameworks** like TensorFlow and PyTorch, but also based on traditional programming model like Pthreads.
- Provides Training, Inference, Micro and Synthetics Benchmarks across Datacenter, HPC, IoT, and Edge.

CONS

- Last update in 2021
- **Lacking benchmark results** on latest Mobile and CPU/GPUs architectures

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017142

4.4. OpenML CC-18

4.4.1. Scope

Seamlessly integrated into the OpenML platform, this benchmark suites standardize the setup, execution, analysis, and reporting of benchmarks. Moreover, they make benchmarking a whole lot easier:

- All datasets are uniformly formatted in standardized data formats.
- They can be easily downloaded programmatically through APIs and client libraries.
- They come with machine-readable meta-information, such as the occurrence of missing values, to train algorithms correctly.
- Standardized train-test splits are provided to ensure that results can be objectively compared.
- Results can be shared in a reproducible way through the APIs.
- Results from other users can be easily downloaded and reused.

Reference Paper: https://arxiv.org/pdf/1708.03731.pdf

Website: https://www.openml.org/search?type=benchmark&sort=tasks_included&study_type=task, Benchmarking Doc: https://docs.openml.org/benchmark/

4.4.2. Metrics

The OpenML-CC18 contains all verified and publicly licenced OpenML datasets until mid-2018 that satisfy a large set of clear requirements for thorough yet practical benchmarking:

- The number of observations is between 500 and 100000 to focus on medium-sized datasets that can be used to train models on almost any computing hardware.
- The dataset has less than 5000 features, counted after one-hot-encoding categorical features (which is the most frequent way to deal with categorical variables), to avoid most memory issues.
- The target attribute has at least two classes, with no class of less than 20 observations. This ensures sufficient samples per class per fold when running 10-fold cross-validation experiments.
- The ratio of the minority and majority class is above 0.05 (to eliminate highly imbalanced datasets which require special treatment for both algorithms and evaluation measures).
- The dataset is not sparse because not all machine learning models can handle them gracefully, this constraint facilitates our goal of wide applicability.
- The dataset does not require taking time dependency between samples into account, e.g., time series or data streams, as this is often not implemented in standard machine learning libraries. Removed datasets where each sample constitutes a single data stream.
- The dataset does not require grouped sampling. Such datasets would contain multiple data points for one subject and require that all data points for a subject are put into the same data split for evaluation.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017142

4.4.3. Pros/Cons

PROS

- **Easy creation of benchmarks**
- **Permanence and provenance**: Because benchmarking suites are its own entity on OpenML, it is clear who created them (provenance).
- **Community of practice**: Curated benchmark suites allow scientists to thoroughly benchmark their machine learning methods without having to worry about finding and selecting datasets for their benchmarks.
- **Building on existing suites**: Scientists can extend, subset, or adapt existing benchmarking suites to correct issues, raise the bar, or run personalized benchmarks.
- **Reproducibility of benchmarks**: Based on machine-readable OpenML tasks, with detailed instructions for evaluation procedures and train-test splits, shared results are comparable and reproducible.

CONS

- **Overfitting**: overfitting on fixed suites is increasingly likely.
- **Computational issues**: focused on mid-size datasets, some larger ones still incurred too high computational load, so some researchers have used subsets of the OpenML-CC18 in their work
- **Breadth of current benchmarking suites**: researchers are interested in benchmarking larger (deep learning) models on larger datasets from many domains (including language and vision).
- **Specification of resource constraints**: the task and suite specifications do not yet allow for constraints on resources, e.g., memory or time limits.

4.5. DataPerf

4.5.1. Scope

DataPerf has the following goals:

- Focus research and development on improving ML dataset quality.
- Improve ML training datasets to increase accuracy and/or reduce data required to train.
- Improve ML test datasets to drive ML solution fidelity and reliability.
- Motivate datasets that increase representation and decrease bias.
- Drive development of better techniques and tools for creating and optimizing datasets.
- Provide consistent metrics for researchers and commercial developers.
- Enforce replicability to ensure reliable results.
- Keep benchmarking effort affordable so all can participate.

Reference Paper: https://arxiv.org/pdf/2207.10062

Website: https://dataperf.org/

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017142

4.5.2. Metrics

The DataPerf suite includes the benchmark types listed below. Each benchmark type uses a different metric, though all in principle either maximize the efficacy of a training set or the breadth/difficulty of a test set.

- Training dataset: create a novel training dataset that maximizes the accuracy of a standard set of models trained on it.
- Test dataset: identify novel test data which is incorrectly labeled by the maximum percentage of a standard set of trained models yet is correctly labeled by humans.
- Selection algorithm: select a subset of a larger dataset for use as a training set to maximize the accuracy of a standard set of models trained on it.
- Debugging algorithm: identify mislabeled or unlabeled data that, when corrected, maximizes the accuracy of a standard set of models trained on it.
- Slicing algorithm: identify semantically consistent slices of a dataset for which a trained model underperforms and maximize top-K precision of such identification vs. ideal choices.
- Valuation: estimate the increase in accuracy from supplementing a known training dataset with new data, presently unlabeled, and minimize the error with respect to the actual value.

4.5.3. Pros/Cons

PROS

- Data parsing
- Data augmentation
- Representation selection
- Data quality assessment
- Data acquisition
- Data cleaning

CONS

- Under Development, not publicly available

5. Discussion and alignment with AI-on-demand platform

5.1. Benchmarks discussion

Multiple insights could be gained from the survey of available datasets and benchmarks presented in the previous sections. In particular, we will consider the relation of the benchmarks with the final **goal of the WP6**, which is to **build a vertical matchmaking component** capable of matching available hardware resources and AI algorithms while respecting user-specified constraints. One of the critical components of the vertical engine is a Machine Learning model, which learns the relation between the behaviour of the algorithm (for instance, represented as time-to-solution, runtime, memory consumption, power consumption, and other similar metrics) and the HW platform used to run the algorithm.

From the point of view of the vertical matchmaking engine, most of the value of the benchmarking data lies in the possibility of characterizing the behaviour of AI algorithms running across different hardware platforms

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017142

and under different configurations. The main key to be drawn is to understand whether the deployment of the algorithm can be maximized by being deployed on a specific platform or configuration. The vertical matchmaker will then use this information to support the user in selecting the optimal hardware resources for a given task, i.e., a specific algorithm, and possibly to find the optimal hyperparameters' configuration.

The **benchmarks provided by StairwAI**, e.g., LPDNN's dataset, have into consideration the support of multiple platforms and configurations, e.g., inference engine, model version, model size, data type, etc., **addressing those vertical matchmaking engine's requirements** explained above. Thus, the **first version of vertical matchmaking has already been trained** on this dataset, which will be further described in D6.5.

However, this information is not always explicitly reflected in **the external benchmarking datasets**, which are not always created to characterize the algorithms' behaviour over many different HW platforms, i.e., they tend to collect information about runs executed on a single device. This does not necessarily make these data worthless, as helpful information can be extracted, especially for transfer learning purposes.

It can also be noticed that there is **no general unified format employed** to collect data and perform the benchmarks. This lack of a standard hinders the development of a vertical matchmaking engine over different domains, as an ad-hoc technique to process the benchmarking data would need to be implemented for each case. A standard and unified format could be highly beneficial in this regard – it could be as simple as deciding a common format by which the collected data should be organized. For instance, the first step could consist in creating a companion meta-data descriptor containing the information about the benchmarking data and easily readable by a machine for automated processing.

Another observation that can be made is the fact that existing benchmarking data sets cover a wide range of different target metrics, that is, the metrics used to characterize the algorithms' behaviour and measured during the execution on a specific HW platform. On the one hand, the **variety of target metrics** is a boon as it allows for studying an algorithm's behaviour under different aspects. On the other hand, this might **limit the possibility of comparing** different algorithms and/or different HW devices, as the comparison is difficult when the benchmarks measure different things. Further, fairness can be at stake if the benchmarking process across the various frameworks is not performed homogeneously.

In conclusion, we have generated benchmark datasets using StairwAI's benchmark software, which addresses the heterogeneity of HW devices and have been validated by training the first version of the vertical matchmaking engine. Besides, we have identified existing external benchmark datasets which should be explored more in detail. Further efforts will be put into understanding whether a common format for the external benchmarks dataset can make them useful to extend StairwAI's own datasets and further extend the vertical matchmaking engine's capabilities. If this is the case, other points, such as fairness across the data, should be addressed and validated.

5.2. Alignment with AI-on-demand platform

During the course of this tasks, StairwAI has also participated in an AI4Europe workshop about benchmarking. The main goal of this meeting was to align the efforts for benchmarking across multiple ICT49 projects and the AI-on-demand platform.

During the workshop, multiple benchmarking frameworks and benchmark results were presented. The alignment of benchmark frameworks seemed to be less important as they tend to focus on different AI problems, e.g., training or inference, cloud or edge. Nonetheless, and as concluded in Section 5.1, it happens to be fairly important to align the benchmark results on some established standard format to have a clear common understanding and smooth transition across different AI Tasks.

More benchmark workshops are planned to happen during the extension of StairwAI project to further work on this direction.

6. Conclusion and future work

This document has presented the first version of the *Benchmark results* proposed within WP6 Task 6.2 of StairwAI project. The document has described the benchmark results obtained using Stairwai's benchmark framework from D6.1 across multiple heterogeneous platforms. Besides, we have provided a survey of external benchmarks on the domain of AI to analyse further benefits by combining the two sources of data.

As future work for the 2nd release of the *Benchmark results,* we will include the results of benchmarking Huawei's embedded platform, which will be integrated into LPDNN in T6.1, increasing the capabilities for profiling AI Algorithms on a wide range of platforms.

In addition, we will give more details on the feasibility and results of combining both StairwAI and external benchmarks for the training of the Vertical Matchmaking Engine in T6.3.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017142