|  |  |
| --- | --- |
|  | Moving Picture, Audio and Data Coding by Artificial Intelligence  www.mpai.community |

|  |  |
| --- | --- |
| **Public Document** | |
| N37 | 2020/10/21 |
| Source | MPAI-1 |
| Title | MPAI Application Note #4 - MPAI-AIF – Artificial Intelligence Framework |
| Target | MPAI Members |

### MPAI Application Note #4

### Artificial Intelligence Framework (MPAI-AIF)

**Proponents**: Andrea Basso.

**Description**: The purpose of the MPAI framework is to enable the creation and automation of mixed ML-AI-DP processing and inference workflows at scale. The key components of the framework should address different modalities of operation (AI, ML and DP), data pipeline jungles and computing resource allocations including constrained hardware scenarios of edge AI devices.

The framework is depicted in *Figure 1*. It is composed by:

1. Data Storage component
2. Orchestrator
3. Processing modules (PM), traditional or ML algorithms

![Diagram

Description automatically generated]()

*Figure 1 – MPAI Framework*

1. *MPAI Processing Modules (PM)*

PMs are composed by 4 components as indicated in *Figure 1*:

1. The processing element PM (ML or Traditional Data Processor)
2. Interface to the common data storage format
3. input interfaces
4. output interfaces
5. *Orchestrator*

The PMs in *Figure 1* need to be executed and orchestrated, so that are being executed in the correct order and the timing where needed is respected e.g. that inputs to a PM are computed before the PM is executed. As reported in [6] the glue code between PMs of a complex ML system is often brittle and that custom scripts don’t scale beyond specific cases.

Therefore, it is important to have a specific orchestrator component that supports the implementation of several MPAI application scenarios and ease the efficient usage of PMs.

As shown in figure 1 the orchestrator is characterized by:

1. Interface to PMs
2. Interface to the input
3. Interface to the common data storage format

Note that the Orchestrator should be able to handle from simple orchestration tasks (i.e. represented by the execution of a script) to much more complex tasks with a topology of networked PMs that need to be synchronized according to a given time base.

1. *Data Storage*

The Data Storage component encompass traditional storage and is referred to a variety of data types. Some examples: it stores the inputs and outputs of the individual PMs, data from the orchestrator PM’s state and intermediary results, shared data among PMs, information used by the orches­trator, orchestrator procedures as well domain knowledge data, data models, etc.

**Comments**:

**Examples**

#### MPAI-CAE

Examples of PMs in the MPAI-CAE application scenario are

* process the signals from the environment captured by the microphones
* performing dynamic signal equalization
* selectively recognize and allow relevant environment sounds (i.e. the horn of a car).

#### MPAI-MMC

Examples of PMs in the MPAI-MMC scenario are

* Speech recognition
* Speech synthesis
* Emotion recognition
* Intention understanding
* Gesture recognition
* Knowledge fusion from different sources such as speech, facial expression, gestures, etc

An illustrative application of the MPAI-AIF Framework is given below

![Timeline

Description automatically generated]()

*Figure 2 – MPAI-MMC in the MPAI-AIF Framework*

**Requirements:**

MPAI has identified the following initial requirements:

* Shall allow to orchestrate and automate the execution of AI mixed workflows for a variety of application scenarios
* Agnostic to AI, ML or DP technology: The architecture should be general enough avoiding the imposition of limitations in terms of algorithmic structure, storage and communication and allow full interoperability of its components.
* Allow uninterrupted functionality while algorithms or ML models are updated or retained. ML models once deployed may need to update often as more data is made available through their usage. Deployment of the updated models should happen seamlessly with no or minimal impact.
* Support for distributed computing including combination of cloud and edge AI components.
* Scalable: can be used in scenarios of different complexity
* Efficient use of the computing and communication resources e.g. by supporting task sharing also for ML processing modules.
* Support a common interface for processing modules
* Support common data representation for storage
* Support parallel and sequential combination of PMs
* Support real time processing

**Object of standard**:

MPAI has identified the following areas of possible standardization:

* Architecture that specifies the roles of the components of the architecture
* The modalities of the creation of the pipelines of PMs
* Common PM interfaces specified in the application standards s8ch as MPAI-CAE, etc ..
* Data storage interfaces

The current framework has been made as general as possible taking into consideration also current MPAI application standards. We are expecting the architecture to be enriched and extended according to the proposals of the MPAI contributors.

**Benefits:**

MPAI-AIF will bring benefits positively affecting

1. Technology providers need not develop full applications to put to good use their technol­ogies. They can concentrate on improving the AI technologies of the PMs. Further, their technologies can find a much broader use in application domains beyond those they are accustomed to deal with.
2. Equipment manufacturers and application vendors can tap from the set of technologies made available according to the MPAI-AIF standard from different competing sources, integrate them and satisfy their specific needs
3. Service providers can deliver complex optimizations and thus superior user experience with minimal time to market as the MPAI-AIF framework enables easy combination of 3rd party components from both a technical and licensing perspective. Their services can deliver a high quality, consistent user audio experience with minimal dependency on the source by selecting the optimal delivery method
4. End users enjoy a competitive market that provides constantly improved user exper­iences and controlled cost of AI-based products.

**Bottlenecks:** the full potential of AI in MPAI-AIF would be limited by a market of AI-friendly processing units and by the introduction of the vast amount of AI technologies into products and services.

**Social aspects**:

MPAI-AIF will enable the availability of a variety of AI-based products at lower price and faster.

**Success criteria**:

MPAI-AIF should create a competitive market of AI-based components expos­ing standard interfaces, processing units available to manufacturers, a variety of end user devices and trigger the implicit need felt by a user to have the best experience whatever the context.

**References**

[1] W. Wang, J. Gao, M. Zhang, S. Wang, G. Chen, T. K. Ng, B. C. Ooi, J. Shao, and M. Reyad, “Rafiki: machine learning as an analytics service system,” Proceedings of the VLDB Endowment, vol. 12, no. 2, pp. 128–140, 2018.

[2] Y. Lee, A. Scolari, B.-G. Chun, M. D. Santambrogio, M. Weimer, and M. Interlandi, “PRETZEL: Opening the black box of machine learning prediction serving systems,” in 13th USENIX Symposium on Operating Systems Design and Implementation (OSDI18), pp. 611–626, 2018.

[3] “ML.NET [ONLINE].” https://dotnet.microsoft.com/apps/machinelearning-ai/ml-dotnet.

[69] D. Crankshaw, X. Wang, G. Zhou, M. J. Franklin, J. E. Gonzalez, and I. Stoica, “Clipper: A low-latency online prediction serving system.,” in NSDI, pp. 613–627, 2017.

[4] S. Zhao, M. Talasila, G. Jacobson, C. Borcea, S. A. Aftab, and J. F. Murray, “Packaging and sharing machine learning models via the acumos ai open platform,” in 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 841–846,

IEEE, 2018.

[5] “Apache Prediction I/O.” https://predictionio.apache.org/.

[6] D.Sculley, G.Holt,D.Golovin,E. Davydov,T.Phillips,D.Ebner,V. Chaudhary,M. Young, J. Crespo, D. Dennison "Hidden technical debt in Machine learning systems Share" on NIPS'15: Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2, December 2015, Pages 2503–2511