TVMProposal

Apache TVM Proposal

We propose to incubate the TVM project the Apache Software Foundation. TVM is a full stack open deep learning compiler stack for CPUs, GPUs, and specialized accelerators. It aims to close the gap between the productivity-focused deep learning frameworks, and the performance- or efficiency-oriented hardware backends.

Background

There is an increasing need to bring machine learning to a wide diversity of hardware devices. Current frameworks rely on vendor-specific operator libraries and optimize for a narrow range of server-class GPUs. Deploying workloads to new platforms – such as mobile phones, embedded devices, and accelerators (e.g., FPGAs, ASICs) – requires significant manual effort. TVM is an end to end deep learning a compiler that exposes graph-level and operator-level optimizations to provide performance portability to deep learning workloads across diverse hardware back-ends. TVM solves optimization challenges specific to deep learning, such as high-level operator fusion, mapping to arbitrary hardware primitives, and memory latency hiding. It also automates optimization of low-level programs to hardware characteristics by employing a novel, learning-based cost modeling method for rapid exploration of program optimizations.

Moreover, there is increasing interest in designing specialized hardware which accelerates machine learning. Towards this goal, TVM introduces VTA, an open source deep learning accelerator as part of its stack. The open source VTA driver and hardware design is a crucial step toward building software support for future ASICs. The TVM-VTA flow acts as a is the great frontier for researchers and practitioners to explore specialized hardware designs.

Rationale

Deep learning compilation will be the next frontier of machine learning systems. TVM is already one of the leading open source projects pursuing this direction.

Specifically, TVM provides infrastructure to use machine learning to automatically optimize deployment of deep learning programs on diverse hardware backends.

VTA: Open Source Hardware Design

TVM also contains open source hardware as part of its stack. The VTA hardware design is a fully open sourced deep learning accelerator that allows us to experiment with compiler, driver, runtime, and execute the code on FPGA. VTA provides a path to target future ASICs, and build software-driven solutions to co-design future deep learning accelerators.

Having an open source hardware design in an ASF project is rare and perhaps unprecedented. We put some of our rationale on why it is necessary for the community.

Deep learning specialized ASICs are going to be at the center of the AI revolution. However, given its early shape, there is no open standard, or even any available information hardware interface that allows an open source software to target to. VTA provides such open source hardware abstraction layer and allows us to build in abstractions that can be effectively used to target other deep learning accelerators.

Moreover, there is an increasing need for co-designing future of machine learning systems with the hardware abstraction. Having a co-designed open source hardware stack along with the software creates a path for this route. In short, we need open-source hardware to build the best open source software.

Finally, we can still view VTA design as “software”, as its source code is written in source description language and can generate “binary” which can run on FPGA and possibly simulators.

Current Status

TVM is open sourced under the Apache License for one and half years. See the current project website (https://tvm.ai/), Github (https://github.com/dmlc/tvm/), as well as TVM Conference (https://sampl.cs.washington.edu/tvmconf/#about-tvmconf)

TVM has already been used in production, some highlights are AWS (Sagemaker Neo), Huawei (AI Chip compilation) and Facebook (mobile optimization). We anticipate the list of adopters to grow over the next few years.

Meritocracy

The TVM stack began as a research project of the SAMPL group at Paul G. Allen School of Computer Science & Engineering, University of Washington. The project is now driven by an open source community involving multiple industry and academic institutions. The project is currently governed by the Apache Way (https://docs.tvm.ai/contribute/community.html). The project now has 15 committers and 6 PMC members, and the list is actively growing. The PMC uses a google group mail-list to vote in new committers/PMC members, which will be moved to private@ after incubation.

The community highly values open collaboration among contributors from different backgrounds.The current contributors come from UW, Berkeley, Cornell, SJTU, AMD, AWS, Huawei, Google, Facebook, NTT, Ziosoft, TuSimple and many other organizations

Community

The project currently has 185 contributors. As per the Apache way, all the discussions are conducted in publicly archivable places.

- Github issues are used to track development activities and RFC.
- The roadmap is public and encourages participation from everyone in the community.
- Discussion forums for general discussions. https://discuss.tvm.ai
- The content of the discourse forum can be considered as a public archive as it is searchable with all the content
- We also created a mail-list archive of the forum, which we will forward to an Apache mail-list after incubation https://groups.google.com/forum/#!forum/tvm-discuss-archive

- See https://tvm.ai/community
- See https://github.com/dmlc/tvm/releases for past releases.

Currently, Github issue serves as dev@ channel. Notably, major features always start from RFCs discussions to encourage broad participation in the community.

The community recognizes potential committers early by bringing contributors as code reviewers and encourages them to participate in code reviews. Code reviews and high-quality code are fundamental to the long-term success of the project. The reviewer mechanism in the community serves a way to highlight this aspect as well as helping the community find good candidates to promote to committers.

Development and Decision Process

See https://docs.tvm.ai/contribute/community.html#general-development-process for the current development guideline. The key points are:
Open public roadmap during development, which turns into release notes Major features start with an RFC, everything happens in public Encourage public discussion via archivable channels Strive to reach a consensus on technical decisions through discussion Moderation from committers and encourage everyone's participation

Example Roadmap: https://github.com/dmlc/tvm/issues/1170
The idea is to keep an active list of roadmaps that can be turned directly into a release note. Public roadmap helps to encourage general participation from all contributors.

Example 1: Recently a major proposal in the community is to bring in a new high-level IR, RFC thread: https://github.com/dmlc/tvm/issues/1673 The pull request: https://github.com/dmlc/tvm/pull/1672 Everyone who participated in the RFC is invited to review the code as well

- Follow up features are proposed as follow up RFCs.

Example 2: Community guideline improvements RFC thread: https://github.com/dmlc/tvm/issues/2017 Slack channel setup as per community suggestion, but still encourage the community to only use it for quick communication and use publicly archived channels for development: https://github.com/dmlc/tvm/issues/2174

Example 3: Python3 timeline proposal RFC thread: https://github.com/dmlc/tvm/issues/1602 Finished with the decision to respect backward compatibility and keep python2 support.

See https://github.com/dmlc/tvm/issues?utf8=%E2%9C%93&q=label%3A%22status%3A+RFC%22+
for a full list of RFCs.

Alignment

TVM is useful for building deep learning deployment solutions. It is perhaps also the first Apache incubator proposal that includes both open source software and hardware system design.

It has the potential to benefit existing related ML projects such as MXNet, Singa, SystemML, and Mahout by providing powerful low-level primitives for matrix operations.

Known Risks

Orphaned products

The project has a diverse contributor base. As an example, the current contributors come from UW, Berkeley, Cornell, SJTU, AMD, AWS, Huawei, Google, Facebook, NTT, Ziosoft, TuSimple and many other organizations We are actively growing this list. Given that the project has already been used in production, there is a minimum risk of the project being abandoned.

Inexperience with Open Source

The TVM community has extensive experience in open source. Three of current six PMC members are already PPMC members of existing Apache (incubating) projects. Over the course of development, the community already has a good way bringing RFCs, discussions and most importantly, welcoming new contributors in the Apache way.

Homogenous Developers

The project has a diverse contributor base. As an example, the current contributors come from UW, Berkeley, Cornell, SJTU, AMD, AWS, Huawei, Google, Facebook, NTT, Ziosoft, TuSimple and many other organizations The community actively seeks to collaborative broadly. The PMC members followed a principle to *only* nominate committers outside their own organizations.

Reliance on Salaried Developers

Most of the current committers are volunteers.

Relationships with Other Apache Products

TVM can serve as a fundamental compiler stack for deep learning and machine learning in general. We expect it can benefit projects like MXNet, Spark, Flink, Mahout, and SystemML.

Documentation

See https://tvm.ai/

Initial Source

https://github.com/dmlc/tvm

We plan to move our repository to https://github.com/apache/incubator-tvm

Source and Intellectual Property Submission Plan

TVM source code is available under Apache V2 license. We will work with the committers to get ICLAs signed.

External Dependencies

We put all the source level dependencies under https://github.com/dmlc/tvm/tree/master/3rdparty

- dmlc-core (Apache2): https://github.com/dmlc/dmlc-core
- dlpack (Apache2): https://github.com/dmlc/dlpack
- HalideIR (MIT): https://github.com/dmlc/HalideIR
- range(Unlicense): https://github.com/agauniyal/rang
- Compiler-RT (BSD)
- LLVM

All of the current he dependencies are stable, which means that the current TVM repo is standalone and main development activities only happen at the TVM repo. The dependencies are periodically updated in the rate about once a month when necessary. For source level dependencies, we will always point to a stable release version for software release in the future.

External Dependencies on DMLC projects

There are three dependencies to dmlc projects in the 3rdparty. The current proposal is to keep the current dependencies in the 3rdparty. We elaborate on the background of these dependencies below:

- dmlc-core: is a minimum module for logging and memory serialization. It is currently used by projects including Apache(incubating) MXNet, TVM, and XGBoost. The project is relatively stable, with around one change a week(most recent changes comes from XGBoost project). TVM's dependency on dmlc-core is minimum and only uses its feature for logging.
- dlpack: is a minimum consensus standard for in-memory Tensor format. It is currently used by PyTorch, ApacheMXNet, Chainer, and a few other projects.
- HalideIR: is a minimum IR data structure that is isolated from a fork of Halide project. We keep the license to be MIT to respect the original license and its origin. A common consensus in the TVM project is that we keep the old derived code in HalideIR (which are stable), and all new developments happen in the TVM repo.

The main reason to propose keep these dependencies are:

- Each of the dependencies has the user and developer community of its own which is larger than the TVM community or different license options (MIT in HalideIR)
- These dependencies are stable and update at a monthly rate.

While it is possible to fork the code in the tvm repo, given that the current tvm repo is self-contained, and community development is stand-alone, we feel that there are have enough justifications to treat these as 3rdparty dependencies.

Required Resources

Mailing List:

The usual mailing lists are expected to be set up when entering incubation:

- private@tvm.apache.org
- dev@tvm.apache.org , subscribe github issues.

- discuss-archive@tvm.apache.org, Archive the discuss content of the discourse user forum

Currently, we only use issues for developments and encourage community to use discuss forums when possible. As a result, the current github issues serves similar purposes as dev@, so we propose to subscribe github issues to dev@ after incubation.

The current community use https://discuss.tvm.ai/ for general technical and support discussions. The community forum is maintained by PMC members. We propose to continue to use the forum and archive the posts to an Apache mail-list. We already have the mechanism to do so (see https://groups.google.com/forum/#!forum/tvm-discuss-archive)

Git Repositories:

Upon entering incubation, we plan to transfer the existing repo from https://github.com/dmlc/tvm to https://github.com/apache/incubator-tvm.

Issue Tracking:

TVM currently uses GitHub to track issues. Would like to continue to do so while we discuss migration possibilities with the ASF Infra team.

URL:

Current project website: https://tvm.ai/, as we proceed website will migrate to https://tvm.incubator.apache.org and hopefully https://tvm.apache.org

Initial Committers and PMC Members

As the project has already followed the Apache way of development(in terms of meritocracy, community, and archive of public discussion). We plan to transition the current PMC members to PPMC members , and committers to apache committers. There are also ongoing votes and discussions in the current tvm PMC private mail-list about new committers/PMC members(we also invited our tentative mentors as observers to the mail-list). We plan to migrate the discussions to private@ after the proposal has been accepted and bring in the new committers/PPMC member according to the standard Apache community procedure.

Initial PPMC Members (PMC from TVM project and Mentors)

- Tianqi Chen tqchen@apache.org
- Ziheng Jiang ziheng@apache.org
- Yizhi Liu liuyizhi@apache.org
- Thierry Moreau moreau@cs.washington.edu
- Haichen Shen shenhaichen@gmail.com
- Lianmin Zheng lianminzheng@gmail.com
- Markus Weimer
- Sebastian Schelter
- Byung-Gon Chun
- Henry Saputra
- Timothy Chen
- Furkan Kamaci

Initial Committers (Including TVM PMC members)

- Aditya Atluri Aditya.Atluri@amd.com AMD
- Tianqi Chen tqchen@apache.org University of Washington
- Yuwei Hu huyuwei1995@gmail.com Cornell
- Nick Hynes nhynes@berkeley.edu UC Berkeley
- Ziheng Jiang ziheng@apache.org University of Washington
- Yizhi Liu liuyizhi@apache.org AWS
- Thierry Moreau moreau@cs.washington.edu University of Washington
- Jared Roesch jroesch@cs.washington.edu University of Washington
- Siva srk.it38@gmail.com Huawei
- Haichen Shen shenhaichen@gmail.com AWS
- Masahiro Masuda masahi129@gmail.com Ziosoft
- Zhixun Tan phisiart@gmail.com Google
- Leyuan Wang laurawly@gmail.com AWS
- Eddie Yan eqy@cs.washington.edu University of Washington
- Lianming Zheng lianminzheng@gmail.com Shanghai Jiao Tong University

Sponsors:

Champion:

- Markus Weimer <weimer@apache.org>

Mentors:

- Sebastian Schelter, New York University
- Byung-Gon Chun, Seoul National University <bgchun@apache.org>

- Henry Saputra <hsaputra@apache.org>
- Timothy Chen <tnachen@apache.org>
- Furkan Kamaci <furkankamaci@gmail.com>

Sponsoring Entity

We are requesting the Incubator to sponsor this project.