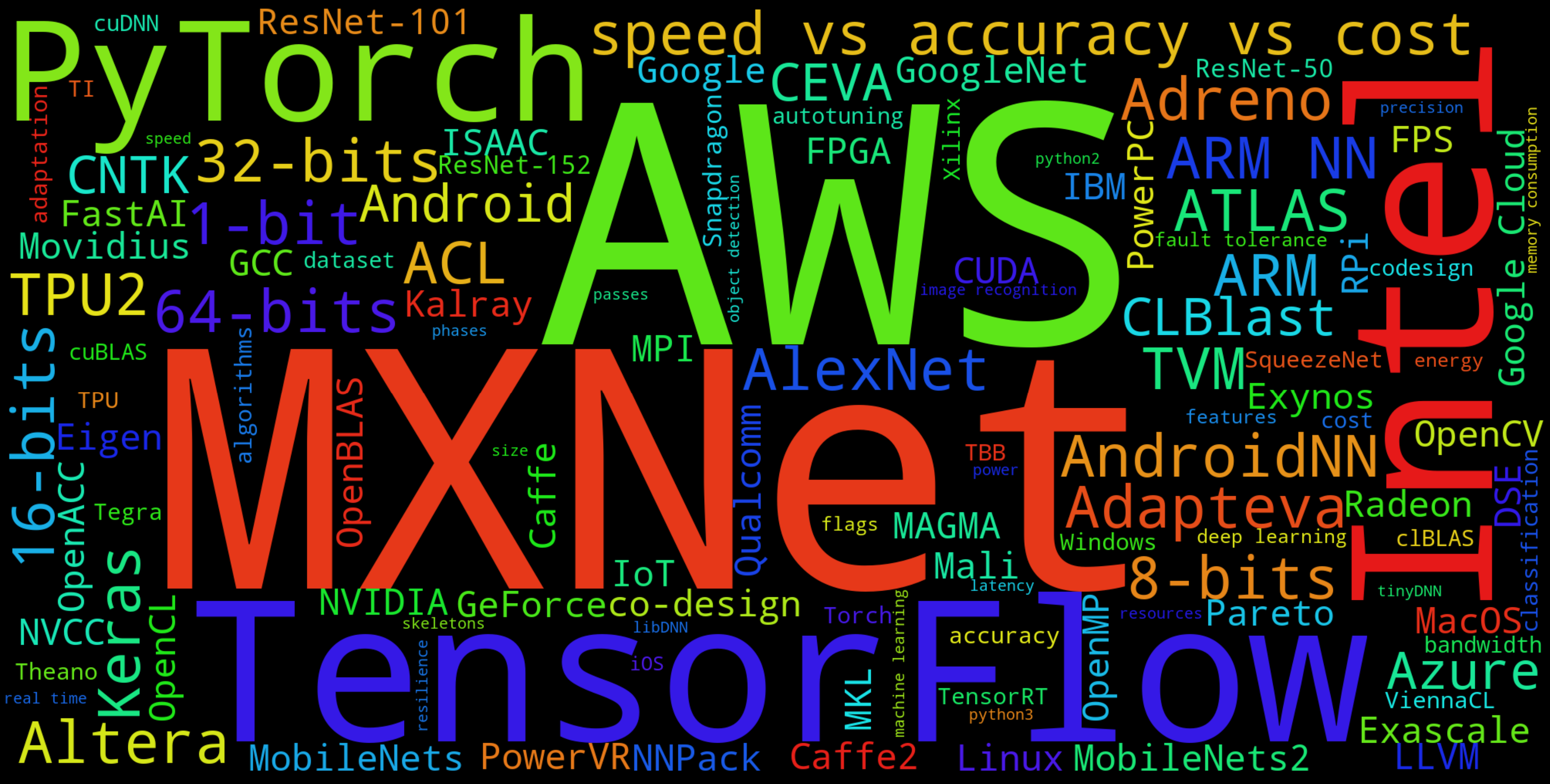
cKnowledge.org: accelerating open science and AI with automated, portable, customizable and reusable research components and workflows



or how to bring some order to R&D chaos and complexity

dividiti.com

Many groups are working to co-design efficient SW/HW stacks for emerging workloads

Many cross-disciplinary R&D groups work on these topics in 2018

Hardware

- All major vendors (NVIDIA, Intel, Google, ARM, Intel, IBM, AMD …)

AI models

Many groups in academia & industry (Google, OpenAI, Microsoft, Facebook …)

AI software

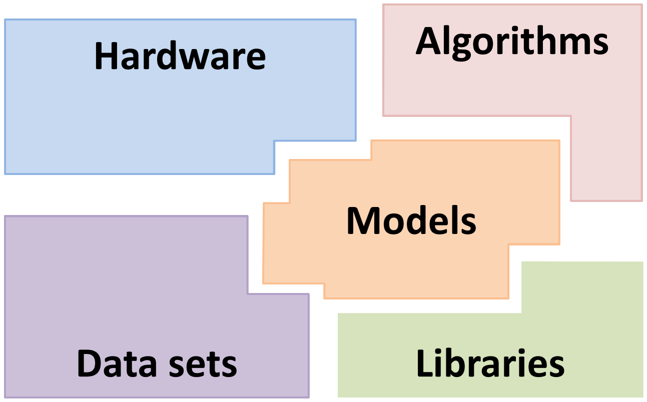
- AI frameworks (TensorFlow, MXNet, PyTorch, CNTK, Theano)
- AI libraries (cuDNN, libDNN, ArmCL, OpenBLAS)

Integration/services

- Cloud services (AWS, Google, Azure ...)

→

Successful computer system must be carefully co-designed for real workloads such as AI and ML



while trading off multiple constraints
(accuracy, speed, energy, size, costs)

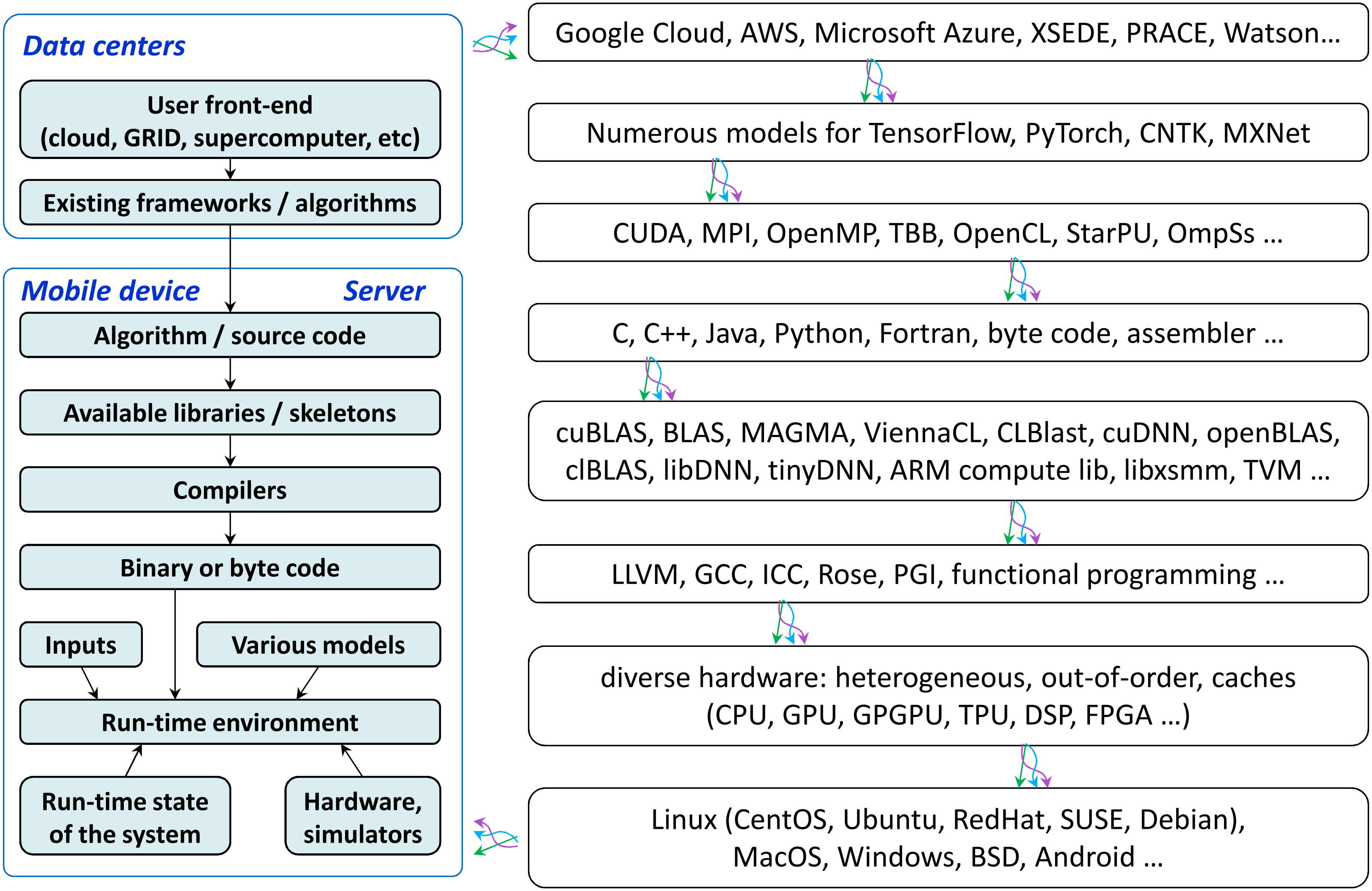
and maximizing ROI
(faster time to market,
R&D sustainability,
much better than all competitors)

→

Helping the society

Healthcare
Agriculture
Finances
Automotive
Aerospace
Meteorology
Retail
Robotics
…

However, system complexity is growing rapidly



Data centers

User front-end (cloud, GRID, supercomputer, etc)

Existing frameworks / algorithms

Mobile device *Server*

Algorithm / source code

Available libraries / skeletons

Compilers

Binary or byte code

Inputs

Various models

Run-time environment

Run-time state of the system

Hardware, simulators

Google Cloud, AWS, Microsoft Azure, XSEDE, PRACE, Watson…

Numerous models for TensorFlow, PyTorch, CNTK, MXNet

CUDA, MPI, OpenMP, TBB, OpenCL, StarPU, OmpSs …

C, C++, Java, Python, Fortran, byte code, assembler …

cuBLAS, BLAS, MAGMA, ViennaCL, CLBlast, cuDNN, openBLAS, clBLAS, libDNN, tinyDNN, ARM compute lib, libxsmm, TVM …

LLVM, GCC, ICC, Rose, PGI, functional programming …

diverse hardware: heterogeneous, out-of-order, caches (CPU, GPU, GPGPU, TPU, DSP, FPGA …)

Linux (CentOS, Ubuntu, RedHat, SUSE, Debian), MacOS, Windows, BSD, Android …

1000+ AI/ML/systems papers published every year with new techniques and solutions

Many cross-disciplinary R&D groups work on these topics in 2018

Hardware

- All major vendors (NVIDIA, Intel, Google, ARM, Intel, IBM, AMD …)

AI models

Many groups in academia & industry (Google, OpenAI, Microsoft, Facebook …)

AI software

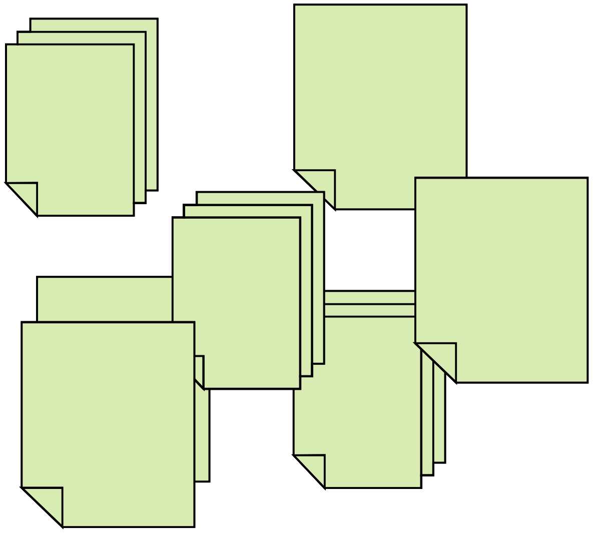
- AI frameworks (TensorFlow, MXNet, PyTorch, CNTK, Theano)
- AI libraries (cuDNN, libDNN, ArmCL, OpenBLAS)

Integration/services

- Cloud services (AWS, Google, Azure ...)

→

Numerous papers, initiatives, tools and events



Numerous available models, data sets, benchmarks, libraries and tools

Public optimization competitions (Kaggle, LPIRC, SCC)

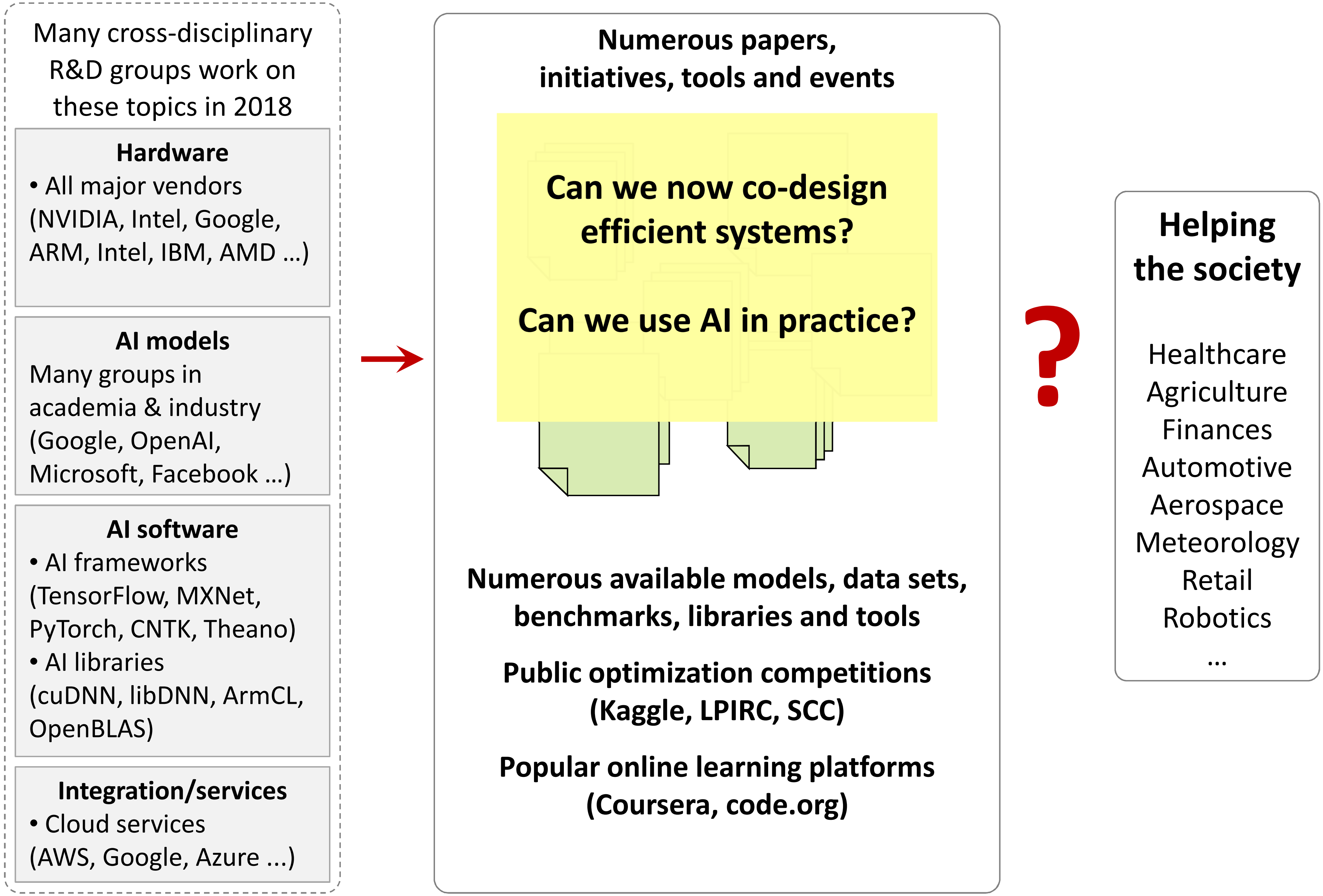
Popular online learning platforms (Coursera, code.org)

?

Helping the society

Healthcare
Agriculture
Finances
Automotive
Aerospace
Meteorology
Retail
Robotics
…

1000+ AI/ML/systems papers published every year with new techniques and solutions



Many cross-disciplinary R&D groups work on these topics in 2018

Hardware

- All major vendors (NVIDIA, Intel, Google, ARM, Intel, IBM, AMD ...)

AI models

Many groups in academia & industry (Google, OpenAI, Microsoft, Facebook ...)

AI software

- AI frameworks (TensorFlow, MXNet, PyTorch, CNTK, Theano)
- AI libraries (cuDNN, libDNN, ArmCL, OpenBLAS)

Integration/services

- Cloud services (AWS, Google, Azure ...)

Numerous papers, initiatives, tools and events

Can we now co-design efficient systems?

Can we use AI in practice?

Numerous available models, data sets, benchmarks, libraries and tools

Public optimization competitions (Kaggle, LPIRC, SCC)

Popular online learning platforms (Coursera, code.org)

?

Helping the society

Healthcare
Agriculture
Finances
Automotive
Aerospace
Meteorology
Retail
Robotics
...

Very few techniques are adopted by industry. They are often not ready due to:

Many cross-disciplinary R&D groups work on these topics in 2018

Hardware

- All major vendors (NVIDIA, Intel, Google, ARM, Intel, IBM, AMD …)

AI models

Many groups in academia & industry (Google, OpenAI, Microsoft, Facebook …)

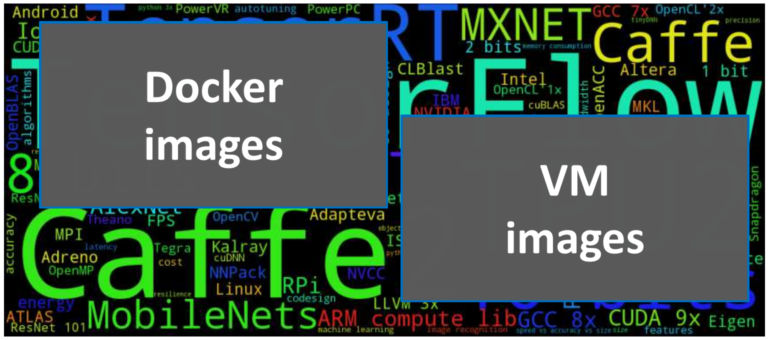
AI software

- AI frameworks (TensorFlow, MXNet, PyTorch, CNTK, Theano)
- AI libraries (cuDNN, libDNN, ArmCL, OpenBLAS)

Integration/services

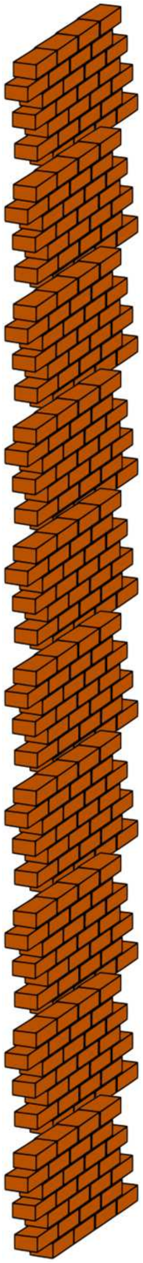
- Cloud services (AWS, Google, Azure ...)

- Technological chaos: continuously changing algorithm/model/SW/HW stack



- Non-representative / outdated training sets
- No common experimental frameworks and established methodologies which can adapt to this chaos
- Numerous reproducibility issues
- Very little artifact reuse
- Very little tech. transfer from academia (toy examples and too many papers)
- Docker and VM images hide the mess but do not solve above problems

Public outcry in academia and industry about research and education crisis



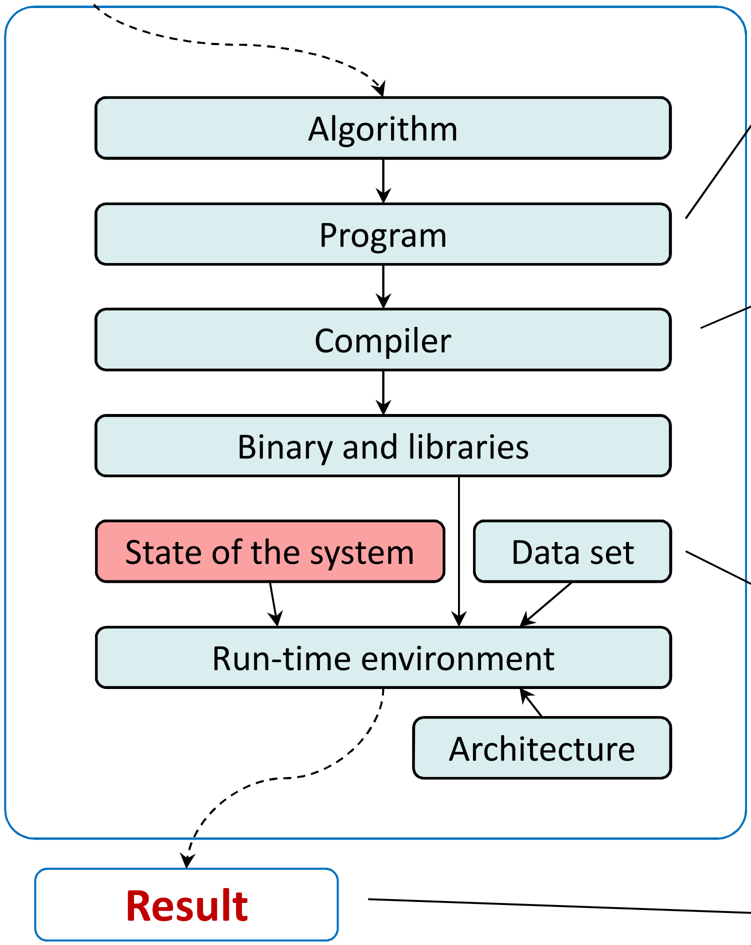
Helping the society

Healthcare
Agriculture
Finances
Automotive
Aerospace
Meteorology
Retail
Robotics
…

Worse: practically all projects develop some soft from scratch to perform similar tasks!

Idea

Creating ad-hoc project with many add-hoc scripts and README files, and sharing it via GitHub, GitLab, ... ***doesn't make it easily portable, reusable and customizable!***



Algorithm

Program

Compiler

Binary and libraries

State of the system

Data set

Run-time environment

Architecture

Result

Ad-hoc scripts to compile and run a program or a benchmark

- image corner detection
- matmul OpenCL
- compression
- neural network CUDA

Have some common meta: *which datasets can use, how to compile, CMD, …*

Ad-hoc scripts to install packages or set up environment for code and data deps on a given platform

- GCC V8.1
- LLVM V7.0
- Intel Compilers 2017

Have some common meta: *compilation, linking and optimization flags*

Ad-hoc dirs for data sets with some ad-hoc scripts to find them, extract features, etc

- image-jpeg-0001
- bzip2-0006
- txt-0012
- video-raw-1280x1024

Have some (common) meta: *filename, size, width, height, colors, …*

Ad-hoc dirs and scripts to record and analyze experiments

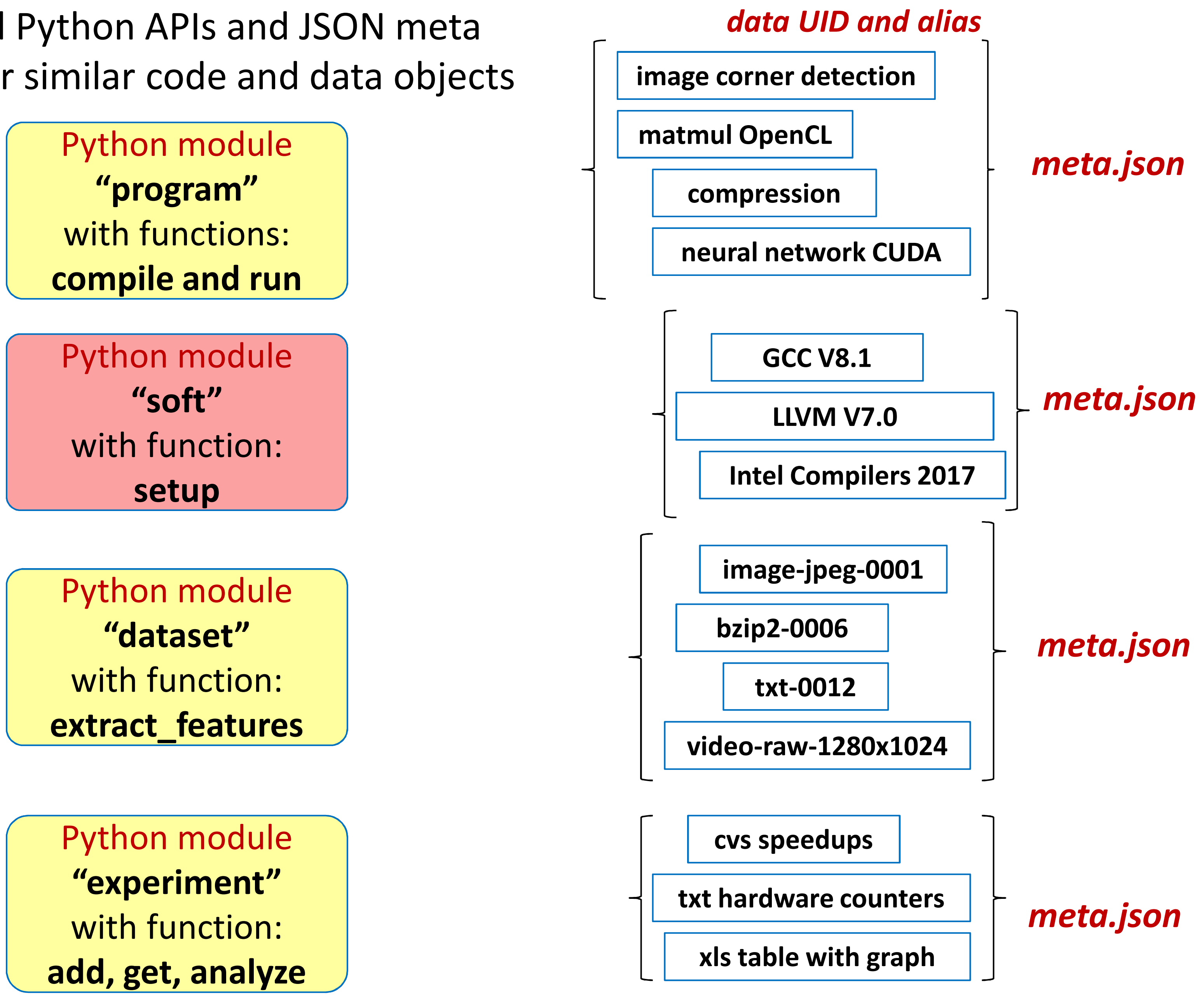
- cvs speedups
- txt hardware counters
- xls table with graph

Have some common meta: *features, characteristics, optimizations*

Very often software from published papers die when students leave or projects finish!

Collective Knowledge concept (CK): share code and data as reusable components

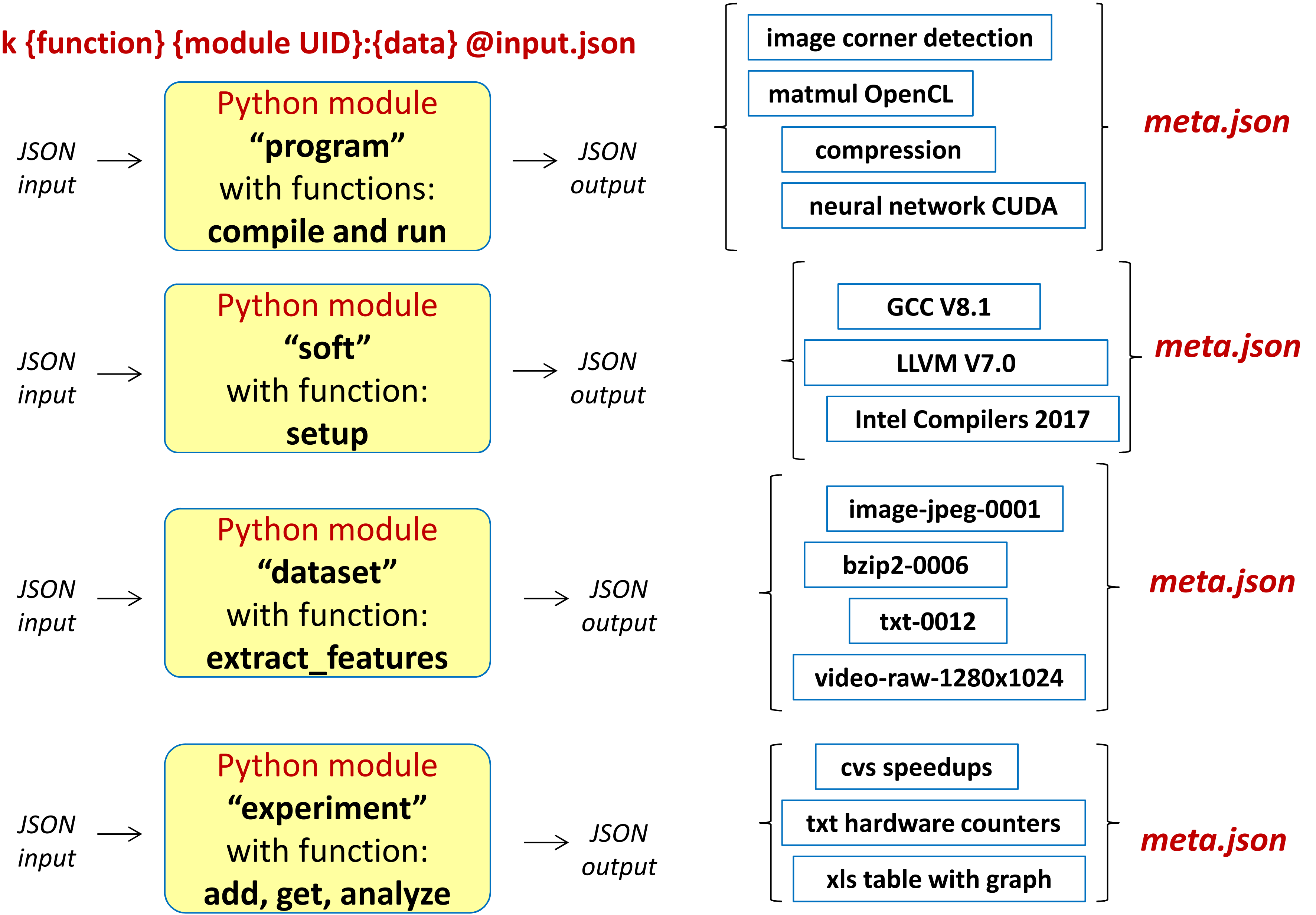
Provide unified Python APIs and JSON meta descriptions for similar code and data objects



CK framework: help users handle reusable research components from command line

CK: small python module (~200Kb); no extra dependencies; Linux; Win; MacOS

```
$ ck {function} {module UID}:{data} @input.json
```



data UID and alias

JSON input → Python module **"program"** with functions: **compile and run** → *JSON output*

image corner detection
matmul OpenCL
compression
neural network CUDA

meta.json

JSON input → Python module **"soft"** with function: **setup** → *JSON output*

GCC V8.1
LLVM V7.0
Intel Compilers 2017

meta.json

JSON input → Python module **"dataset"** with function: **extract_features** → *JSON output*

image-jpeg-0001
bzip2-0006
txt-0012
video-raw-1280x1024

meta.json

JSON input → Python module **"experiment"** with function: **add, get, analyze** → *JSON output*

cvs speedups
txt hardware counters
xls table with graph

meta.json

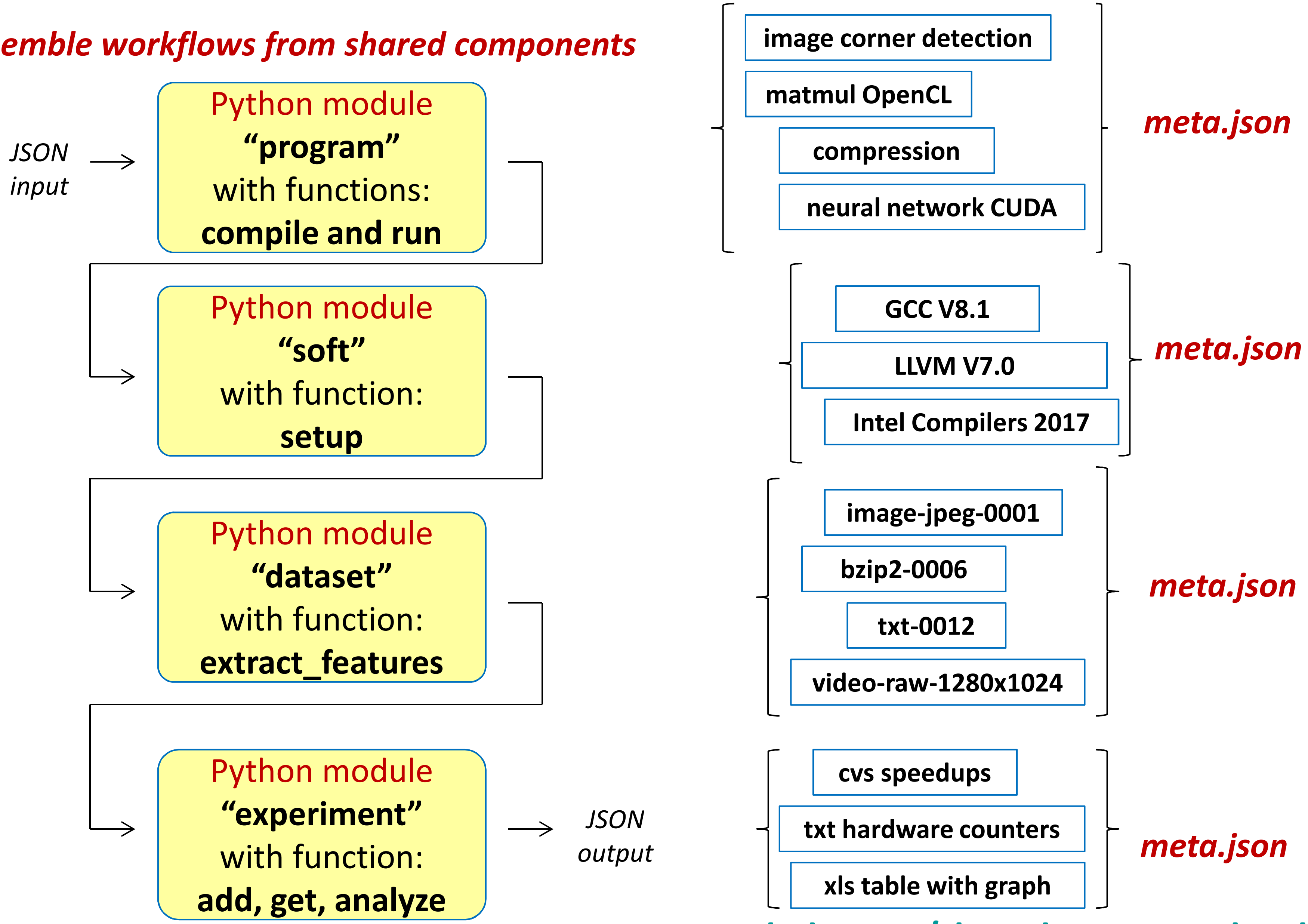
cKnowledge.org/shared-modules.html

CK framework: help users develop research workflows from shared components

CK: small python module (~200Kb); no extra dependencies; Linux; Win; MacOS

Assemble workflows from shared components

data UID and alias



JSON input

Python module **“program”** with functions: **compile and run**

Python module **“soft”** with function: **setup**

Python module **“dataset”** with function: **extract_features**

Python module **“experiment”** with function: **add, get, analyze**

JSON output

image corner detection
matmul OpenCL
compression
neural network CUDA — ***meta.json***

GCC V8.1
LLVM V7.0
Intel Compilers 2017 — ***meta.json***

image-jpeg-0001
bzip2-0006
txt-0012
video-raw-1280x1024 — ***meta.json***

cvs speedups
txt hardware counters
xls table with graph — ***meta.json***

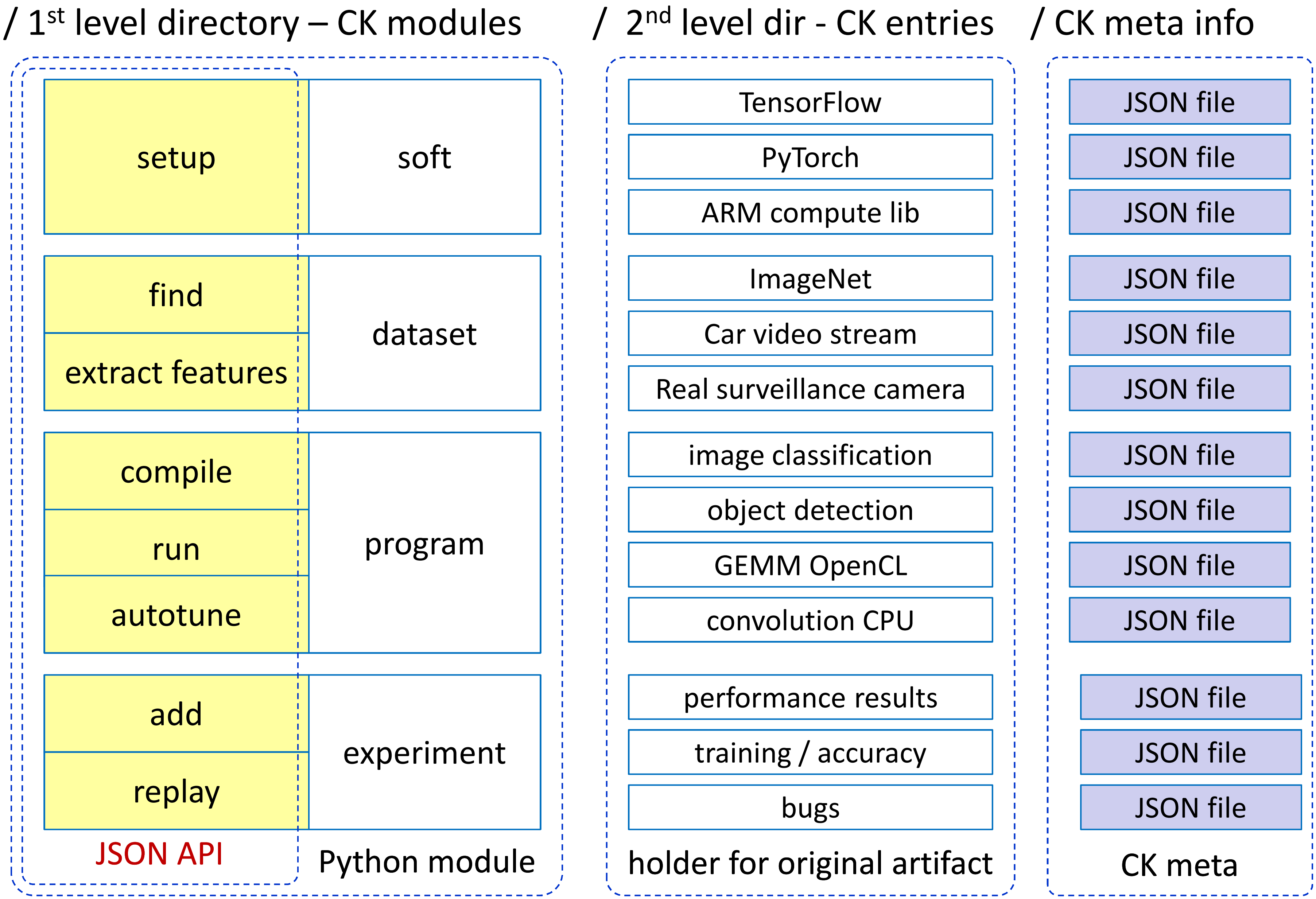
cKnowledge.org/shared-programs.html

CK framework: provide simple and unified directory structure for research projects

CK directory structure

setup	soft	TensorFlow	with some desc.
		PyTorch	with some desc.
		ARM compute lib	with some desc.
find extract features	dataset	ImageNet	with some desc.
		Car video stream	with some desc.
		Real surveillance camera	with some desc.
compile run autotune	program	image classification	with some desc.
		object detection	with some desc.
		GEMM OpenCL	with some desc.
		convolution CPU	with some desc.
add replay	experiment	performance results	
		training / accuracy	
		bugs	

CK framework: provide simple and unified directory structure for research projects



/ 1st level directory – CK modules

/ 2nd level dir - CK entries

/ CK meta info

setup

soft

find

extract features

dataset

compile

run

autotune

program

add

replay

experiment

JSON API

Python module

TensorFlow

PyTorch

ARM compute lib

ImageNet

Car video stream

Real surveillance camera

image classification

object detection

GEMM OpenCL

convolution CPU

performance results

training / accuracy

bugs

holder for original artifact

JSON file

JSON file

JSON file

JSON file

JSON file

JSON file

JSON file

JSON file

JSON file

JSON file

JSON file

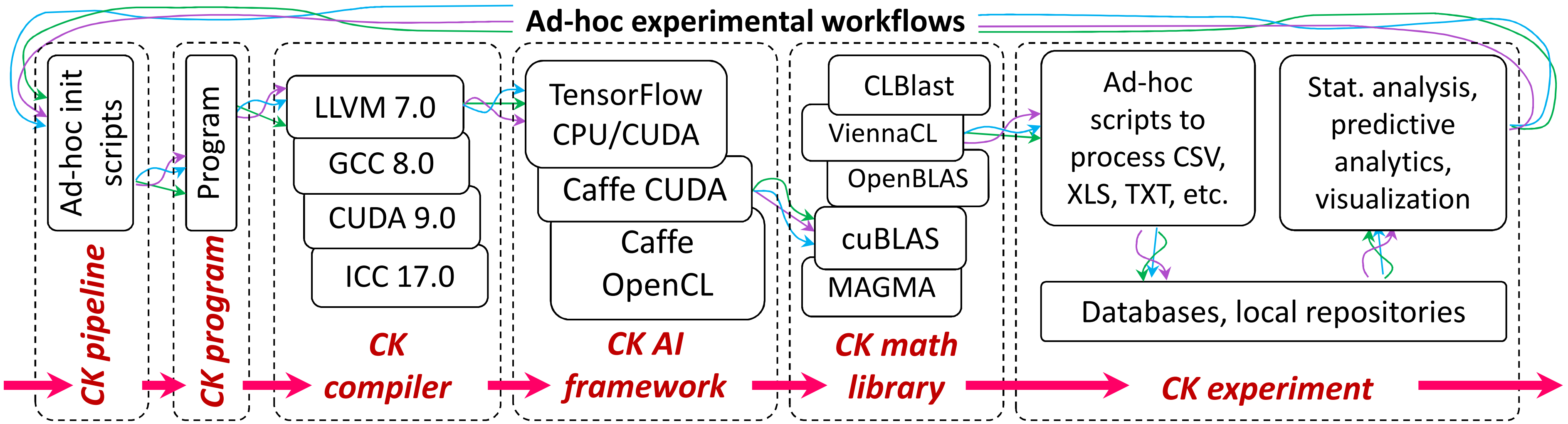
JSON file

JSON file

CK meta

cKnowledge.org/shared-repos.html

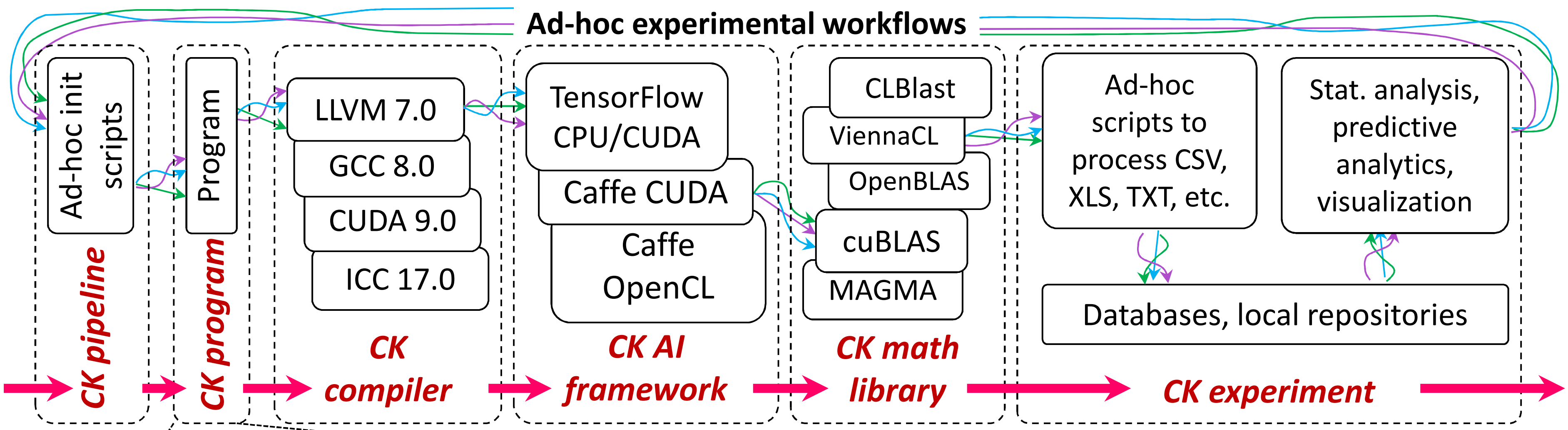
We've already converted multiple AI frameworks, artifacts and programs to the CK



http://cKnowledge.org/shared-programs.html

- github.com/ctuning/ck-scc18
- github.com/ctuning/ck-tensorflow
- github.com/dividiti/ck-caffe

We've already converted multiple AI frameworks, artifacts and programs to the CK



Ad-hoc experimental workflows

Ad-hoc init scripts

CK pipeline

Program

CK program

LLVM 7.0

GCC 8.0

CUDA 9.0

ICC 17.0

CK compiler

TensorFlow CPU/CUDA

Caffe CUDA

Caffe OpenCL

CK AI framework

CLBlast

ViennaCL

OpenBLAS

cuBLAS

MAGMA

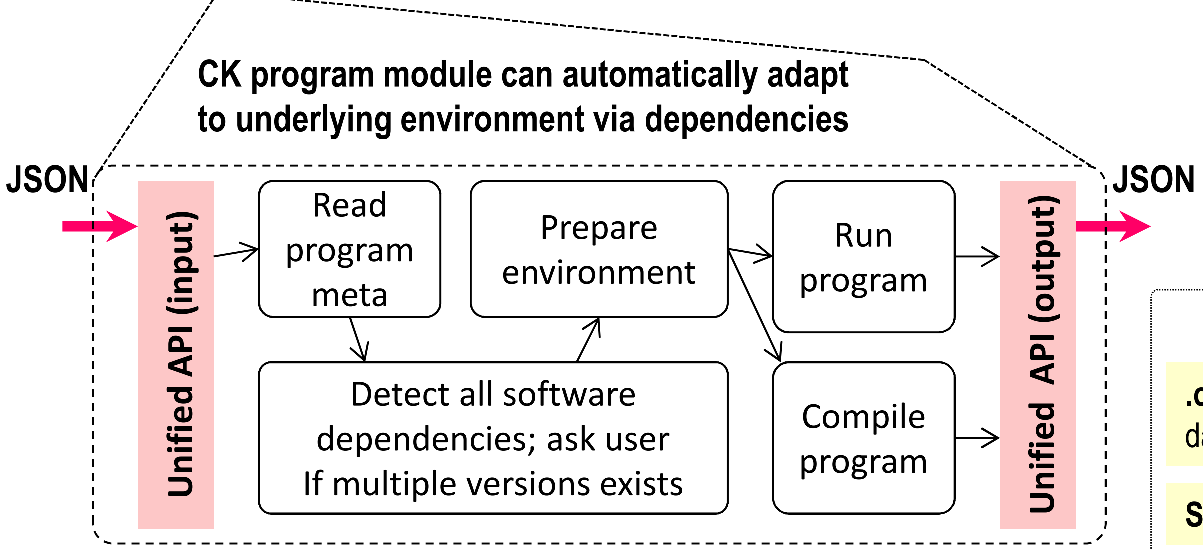
CK math library

Ad-hoc scripts to process CSV, XLS, TXT, etc.

Stat. analysis, predictive analytics, visualization

Databases, local repositories

CK experiment



CK program module can automatically adapt to underlying environment via dependencies

JSON

Unified API (input)

Read program meta

Prepare environment

Detect all software dependencies; ask user If multiple versions exists

Run program

Compile program

Unified API (output)

JSON

CK entries associated with a given module describe a given object **using meta.json** while storing all necessary files and sub-directories

CK program entry *(native directory)*

.cm/meta.json – describes soft dependencies , data sets, and how to compile and run this program

Source files and auxiliary scripts

- github.com/ctuning/ck-scc18
- github.com/ctuning/ck-tensorflow
- github.com/dividiti/ck-caffe

```
$ ck pull repo:ck-scc18
$ ck run program:seissol-proxy
$ ck show env
```

CK framework: automatically adapt workflows to any complex software and hardware

Missing part in many workflows/package managers: detect and plug in already installed software

Soft entries in CK describe how to detect if a given software is already installed, how to set up all its environment including all paths (to binaries, libraries, include, aux tools, etc), and how to detect its version

```
$ ck list soft:compiler*
$ ck search soft --tags=blas
```

```
$ ck detect soft:compiler.gcc
$ ck detect soft --tags=compiler,cuda
$ ck detect soft:compiler.llvm
$ ck detect soft:lib.cublas
```

Env entries are created in CK local repo for all found software instances together with their meta and an auto-generated environment script **env.sh** (on Linux) or **env.bat** (on Windows)

```
$ ck  show env
$ ck show env –tags=cublas
$ ck rm env:* –tags=cublas
```

Local CK repo

local / env / 03ca0be16962f471 / env.sh
Tags: compiler,cuda,v8.0

local / env / 0a5ba198d48e3af3 / env.bat
Tags: lib,blas,cublas,v8.0

Package entries describe how to install a given software if it is not already installed (using CK Python plugin together with **install.sh** script on Linux host or **install.bat** on Windows host). Can be connected with **spack!**

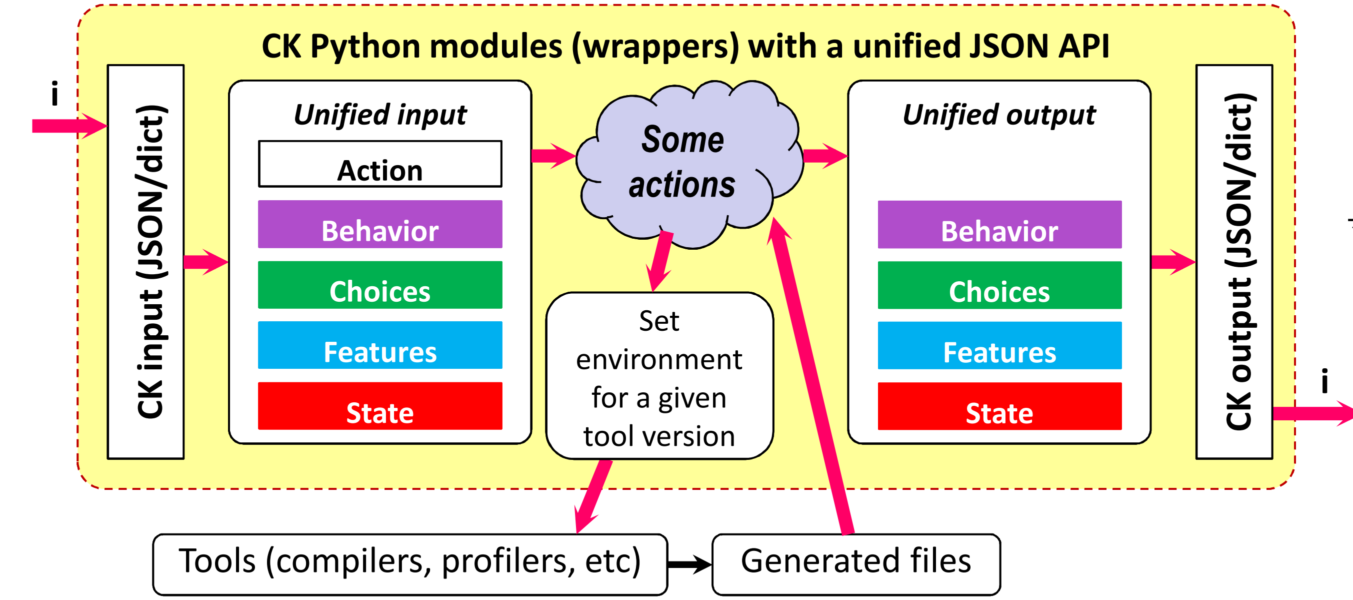
```
$ ck list package:*caffemodel*
$ ck search package –tags=caffe
$ ck list package:*tensorflow*
```

```
$ ck install package:caffemodel-bvlc-googlenet
$ ck install package:imagenet-2012-val
$ ck install package:lib-caffe-bvlc-master-cuda-universal
$ ck install package:lib-tensorflow-cuda
```

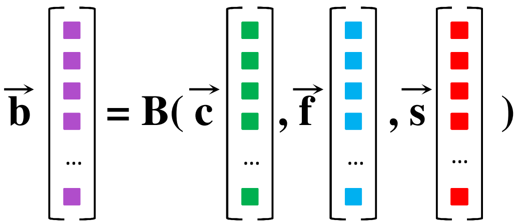
github.com/ctuning/ck/wiki/Portable-workflows

CK workflows: help users autotune the whole AI/ML/SW/HW stack!

First expose coarse grain high-level choices, features, system state and behavior characteristics



Formalized function B of a behavior of any CK object

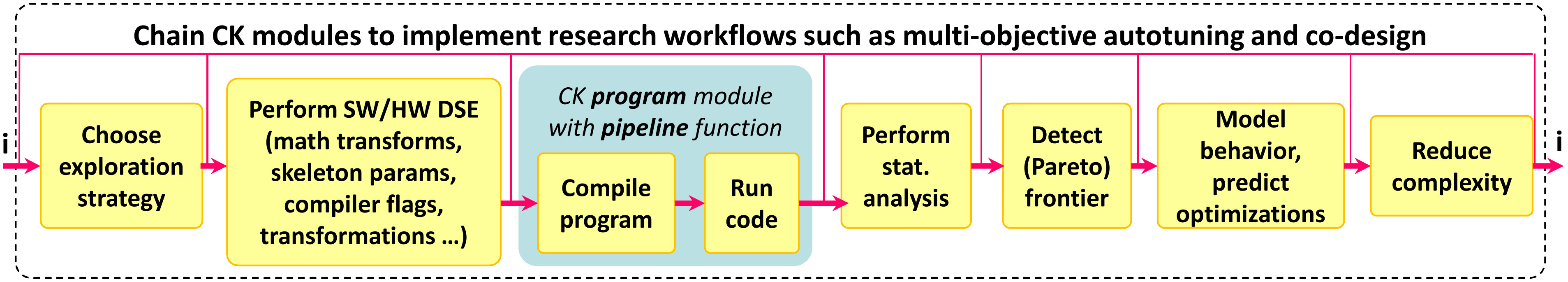


Flattened CK JSON vectors

(dict converted to vector)

to simplify statistical analysis, machine learning and data mining

Crowdsource benchmarking and random exploration across diverse inputs and devices;


Chain CK modules to implement research workflows such as multi-objective autotuning and co-design
i
Choose exploration strategy
Perform SW/HW DSE (math transforms, skeleton params, compiler flags, transformations …)
CK program module with pipeline function
Compile program
Run code
Perform stat. analysis
Detect (Pareto) frontier
Model behavior, predict optimizations
Reduce complexity
i

Keep best species (AI/SW/HW choices); model behavior; predict better optimizations and designs

Collaboration with Raspberry Pi foundation: using CK to teach autotuning and ML:

cKnowledge.org/rpi-crowd-tuning

Gradually expose more information and provide specification (top-down approach)

Autotuning and machine learning specification:

```
{
  "characteristics":{
     "execution times": ["10.3","10.1","13.3"],
     "code size": "131938", ...},
  "choices":{
     "os":"linux", "os version":"2.6.32-5-amd64",
     "compiler":"gcc", "compiler version":"4.6.3",
     "compiler_flags":"-O3 -fno-if-conversion",
     "platform":{"processor":"intel xeon e5520",
                 "l2":"8192“, ...}, ...},
  "features":{
     "semantic features": {"number_of_bb": "24", ...},
     "hardware counters": {"cpi": "1.4" ...}, ... }
  "state":{
     "frequency":"2.27", ...}
}
```

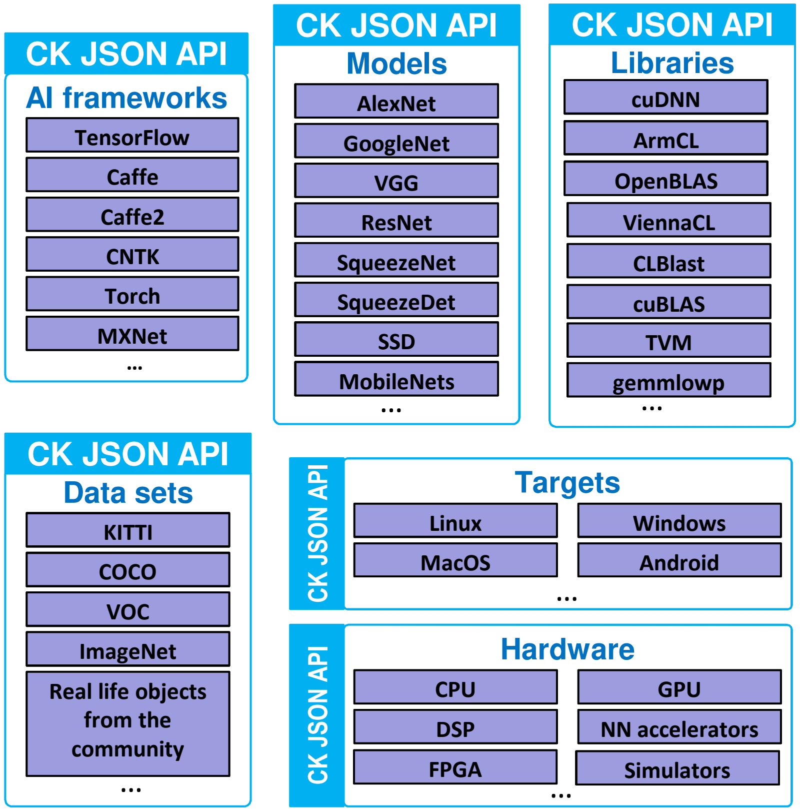
CK flattened JSON key

##characteristics#execution_times@1

```
"flattened_json_key”:{
                  "type": "text”|"integer" | “float" | "dict" | "list”
| "uid",
                  "characteristic": "yes" | "no",
                  "feature": "yes" | "no",
                  "state": "yes" | "no",
                  "has_choice": "yes“ | "no",
                  "choices": [ list of strings if categorical
choice],
                  "explore_start": "start number if numerical
range",
                  "explore_stop": "stop number if numerical
range",
                  "explore_step": "step if numerical range",
                  "can_be_omitted" : "yes" | "no"
                  ...
}
```

CK long-term goal and vision: reboot and accelerate open science

Repositories of customizable, portable and reusable research components with CK API



CK JSON API
AI frameworks
TensorFlow
Caffe
Caffe2
CNTK
Torch
MXNet
…

CK JSON API
Models
AlexNet
GoogleNet
VGG
ResNet
SqueezeNet
SqueezeDet
SSD
MobileNets
…

CK JSON API
Libraries
cuDNN
ArmCL
OpenBLAS
ViennaCL
CLBlast
cuBLAS
TVM
gemmlowp
…

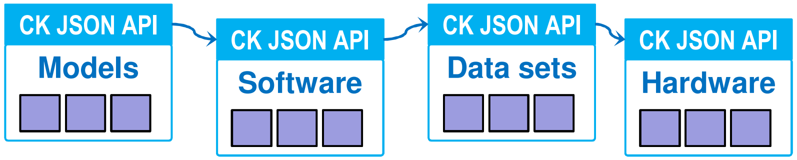
CK JSON API
Data sets
KITTI
COCO
VOC
ImageNet
Real life objects from the community
…

CK JSON API
Targets
Linux
Windows
MacOS
Android
…

CK JSON API
Hardware
CPU
GPU
DSP
NN accelerators
FPGA
Simulators
…

Customizable CK workflows for real-world user tasks

Assemble scenarios such as image classification as LEGO™



CK JSON API
Models

CK JSON API
Software

CK JSON API
Data sets

CK JSON API
Hardware

Share complete workflows along with published papers to automate artifact evaluation and help the community build upon prior work

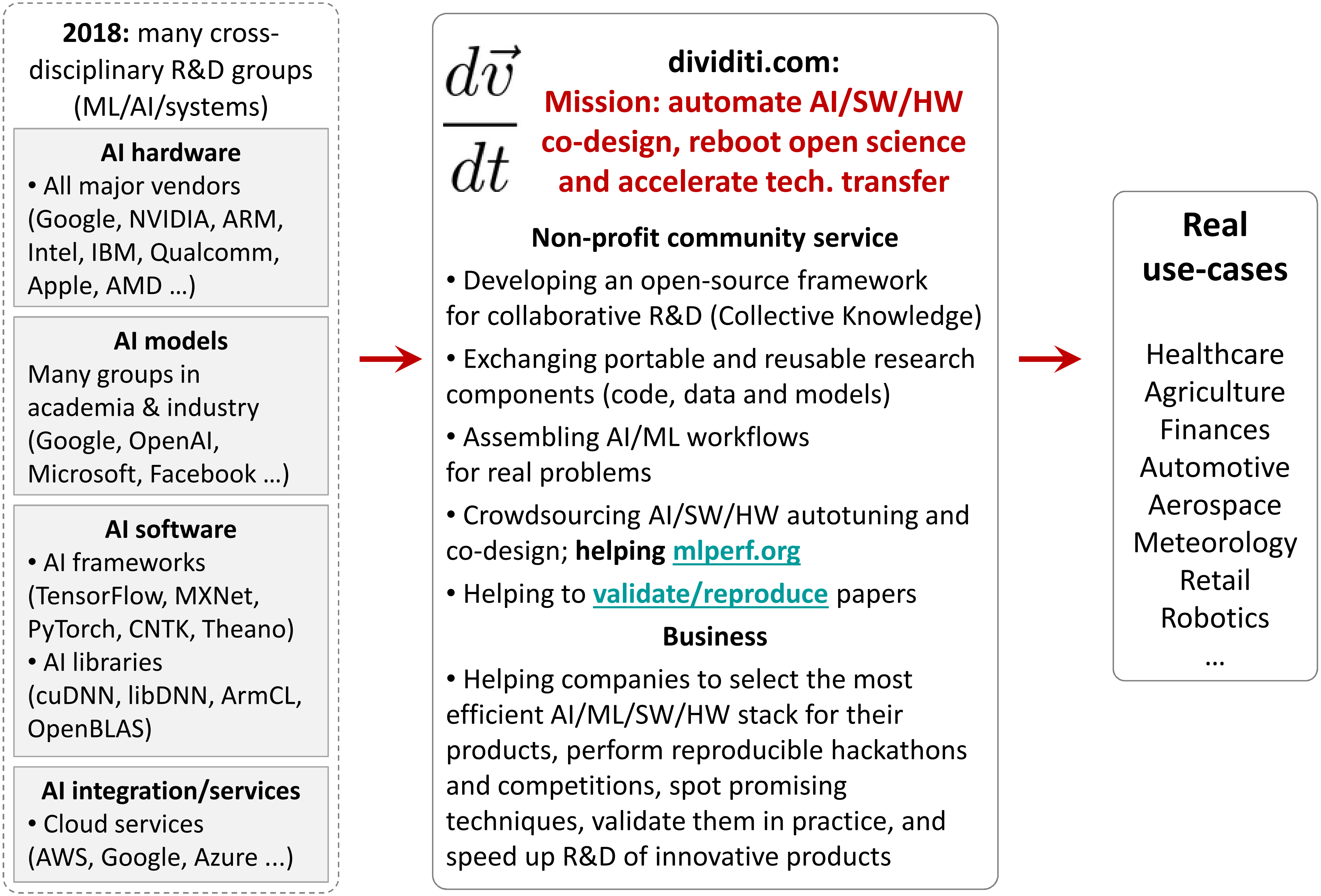
Crowdsource experiments with the help of volunteers across diverse models, data sets and platforms

Present best results, workflows and components on a live scoreboard for fair comparison and reuse
cKnowledge.org/repo

Help students learn multidisciplinary techniques, quickly prototype new ones, validate them in practice with companies, and even contribute back new research components

Help companies select the most appropriate workflows, save R&D costs, accelerate adoption of new techniques!

dividiti: connecting researchers and companies to solve real problems using CK



2018: many cross-disciplinary R&D groups (ML/AI/systems)

AI hardware

- All major vendors (Google, NVIDIA, ARM, Intel, IBM, Qualcomm, Apple, AMD …)

AI models

Many groups in academia & industry (Google, OpenAI, Microsoft, Facebook …)

AI software

- AI frameworks (TensorFlow, MXNet, PyTorch, CNTK, Theano)
- AI libraries (cuDNN, libDNN, ArmCL, OpenBLAS)

AI integration/services

- Cloud services (AWS, Google, Azure ...)

$\frac{d\vec{v}}{dt}$

dividiti.com:

Mission: automate AI/SW/HW co-design, reboot open science and accelerate tech. transfer

Non-profit community service

- Developing an open-source framework for collaborative R&D (Collective Knowledge)
- Exchanging portable and reusable research components (code, data and models)
- Assembling AI/ML workflows for real problems
- Crowdsourcing AI/SW/HW autotuning and co-design; **helping** **mlperf.org**
- Helping to validate/reproduce papers

Business

- Helping companies to select the most efficient AI/ML/SW/HW stack for their products, perform reproducible hackathons and competitions, spot promising techniques, validate them in practice, and speed up R&D of innovative products

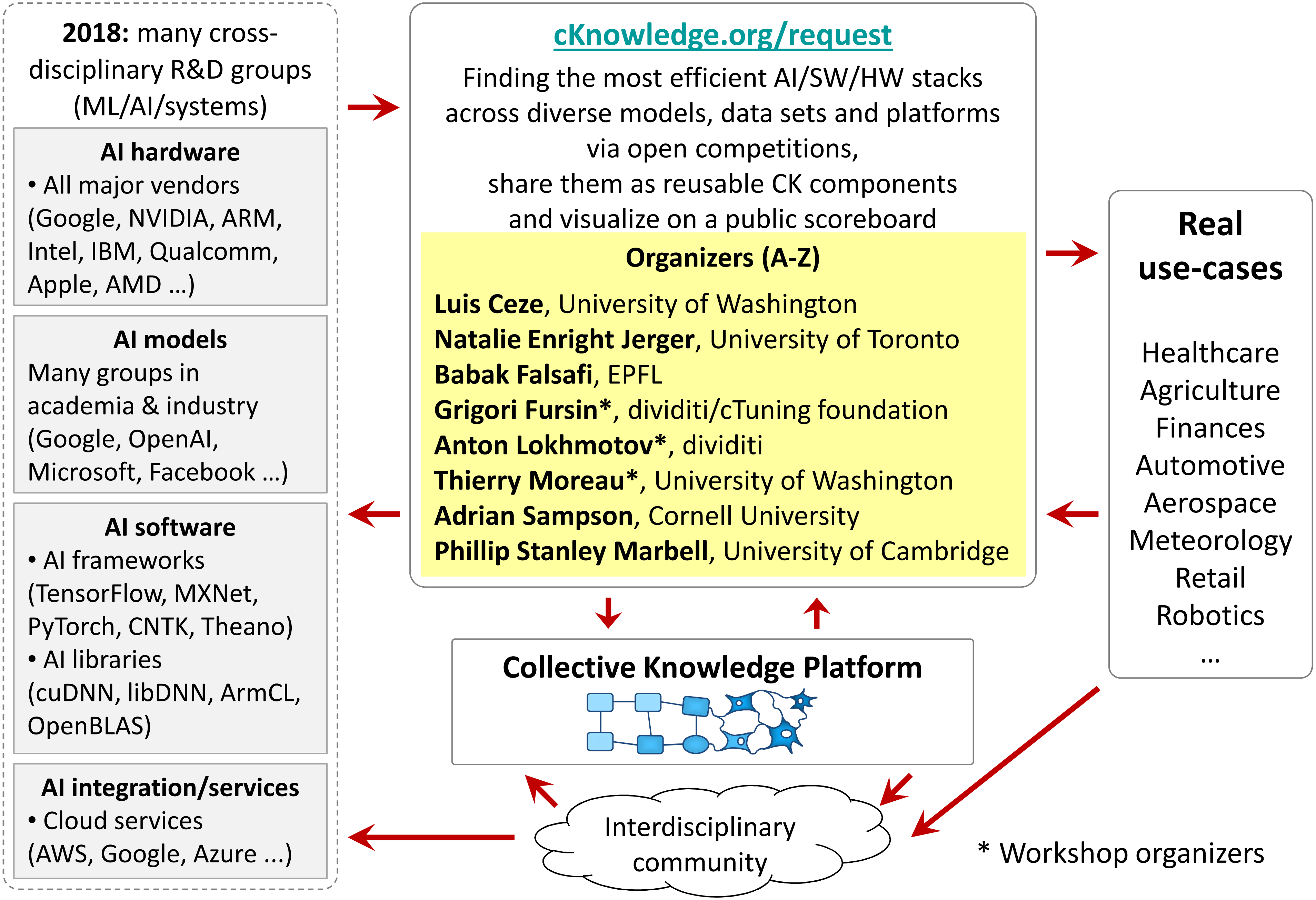
Real use-cases

Healthcare
Agriculture
Finances
Automotive
Aerospace
Meteorology
Retail
Robotics
…

Recent practical use cases of CK as a common research platform

cKnowledge.org/partners

Organizing reproducible tournaments to solve real problems and share all components



2018: many cross-disciplinary R&D groups (ML/AI/systems)

AI hardware

- All major vendors (Google, NVIDIA, ARM, Intel, IBM, Qualcomm, Apple, AMD …)

AI models

Many groups in academia & industry (Google, OpenAI, Microsoft, Facebook …)

AI software

- AI frameworks (TensorFlow, MXNet, PyTorch, CNTK, Theano)
- AI libraries (cuDNN, libDNN, ArmCL, OpenBLAS)

AI integration/services

- Cloud services (AWS, Google, Azure ...)

cKnowledge.org/request

Finding the most efficient AI/SW/HW stacks across diverse models, data sets and platforms via open competitions, share them as reusable CK components and visualize on a public scoreboard

Organizers (A-Z)

Luis Ceze, University of Washington
Natalie Enright Jerger, University of Toronto
Babak Falsafi, EPFL
Grigori Fursin*, dividiti/cTuning foundation
Anton Lokhmotov*, dividiti
Thierry Moreau*, University of Washington
Adrian Sampson, Cornell University
Phillip Stanley Marbell, University of Cambridge

Collective Knowledge Platform

Interdisciplinary community

Real use-cases

Healthcare
Agriculture
Finances
Automotive
Aerospace
Meteorology
Retail
Robotics
…

* Workshop organizers

We formed advisory board (very strong interest from industry)

Advisory/industrial board (A-Z)

- **Michaela Blott**, Xilinx
- **Unmesh Bordoloi**, General Motors
- **Ofer Dekel**, Microsoft
- **Maria Girone**, CERN openlab
- **Wayne Graves**, ACM
- **Vinod Grover**, NVIDIA
- **Sumit Gupta**, IBM
- **James Hetherington**, Alan Turing Institute
- **Steve Keckler**, NVIDIA
- **Wei Li**, Intel
- **Colin Osborne**, ARM
- **Andrew Putnam**, Microsoft
- **Boris Shulkin**, Magna
- **Greg Stoner**, AMD
- **Alex Wade**, Chan Zuckerberg Initiative
- **Peng Wu**, Huawei
- **Cliff Young**, Google

Advisory board suggests algorithms, data sets, models and platforms for competitions.

As a 1st proof-of-concept suggested to build a public repository of the most efficient, portable, customizable and reusable **image classification** algorithms in the CK format optimized across diverse models, data sets and devices from IoT to HPC in terms of accuracy, speed, energy, size, complexity and costs.

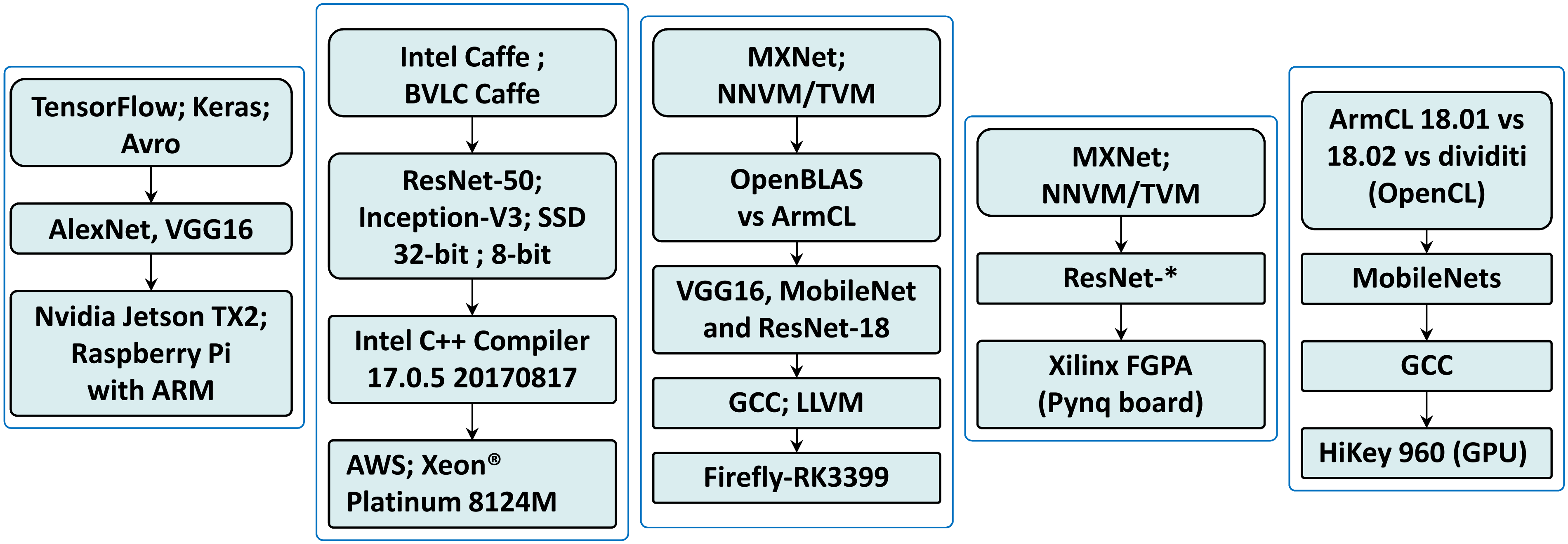
ACM ReQuEST at ASPLOS'18

March 2018

Long term ReQuEST goal: simplify industrial adoption of novel AI/ML/system techniques using common ML/SW/HW co-design framework (CK), realistic workflows and reusable components!

We organized the 1st reproducible tournament at ACM ASPLOS'18

8 intentions to submit and 5 submitted image classification workflows with unified Artifact Appendices



TensorFlow; Keras; Avro

AlexNet, VGG16

Nvidia Jetson TX2; Raspberry Pi with ARM

Intel Caffe ; BVLC Caffe

ResNet-50; Inception-V3; SSD 32-bit ; 8-bit

Intel C++ Compiler 17.0.5 20170817

AWS; Xeon® Platinum 8124M

MXNet; NNVM/TVM

OpenBLAS vs ArmCL

VGG16, MobileNet and ResNet-18

GCC; LLVM

Firefly-RK3399

MXNet; NNVM/TVM

ResNet-*

Xilinx FGPA (Pynq board)

ArmCL 18.01 vs 18.02 vs dividiti (OpenCL)

MobileNets

GCC

HiKey 960 (GPU)

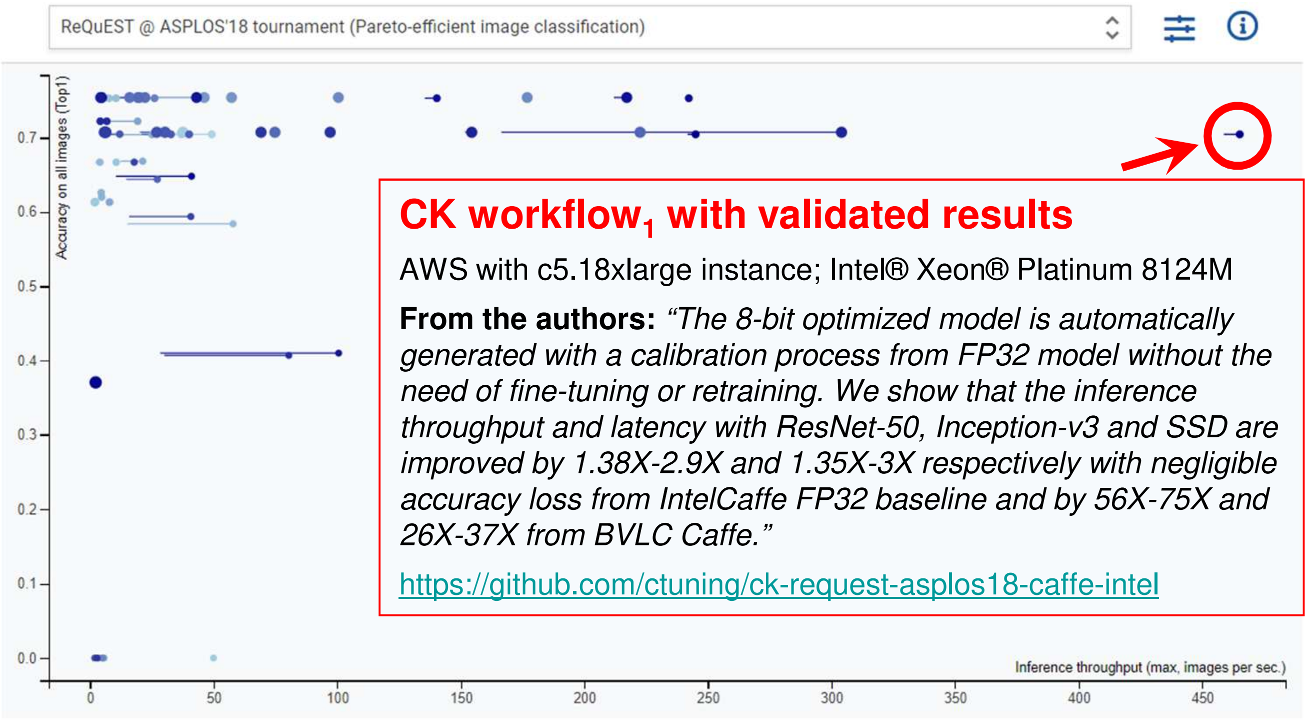
Public validation at github.com/ctuning/ck-request-asplos18-results via GitHub issues.

All validated papers are published in the ACM DL
with all **reusable CK components and workflows**!

See ACM ReQuEST report: portalparts.acm.org/3230000/3229762/fm/frontmatter.pdf

All results and research components are available via a live CK scoreboard

Multi-objective results for all AI/SW/HW stacks are presented on a live scoreboard and become available for public comparison and further customization, optimization and reuse!

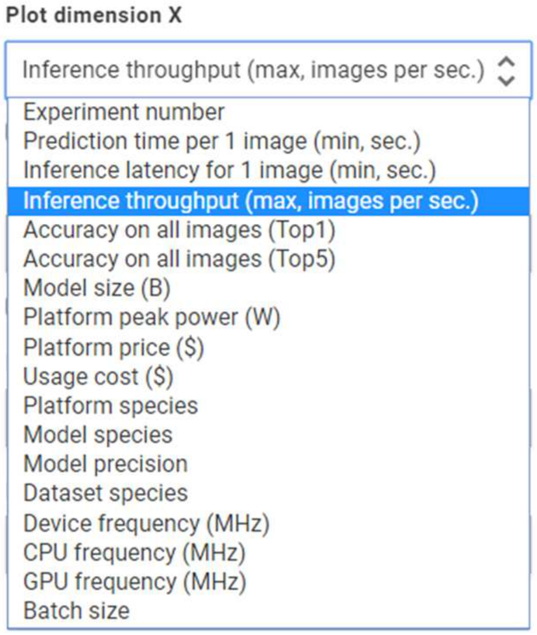


CK workflow$_1$ with validated results

AWS with c5.18xlarge instance; Intel® Xeon® Platinum 8124M

From the authors: *"The 8-bit optimized model is automatically generated with a calibration process from FP32 model without the need of fine-tuning or retraining. We show that the inference throughput and latency with ResNet-50, Inception-v3 and SSD are improved by 1.38X-2.9X and 1.35X-3X respectively with negligible accuracy loss from IntelCaffe FP32 baseline and by 56X-75X and 26X-37X from BVLC Caffe."*

https://github.com/ctuning/ck-request-asplos18-caffe-intel

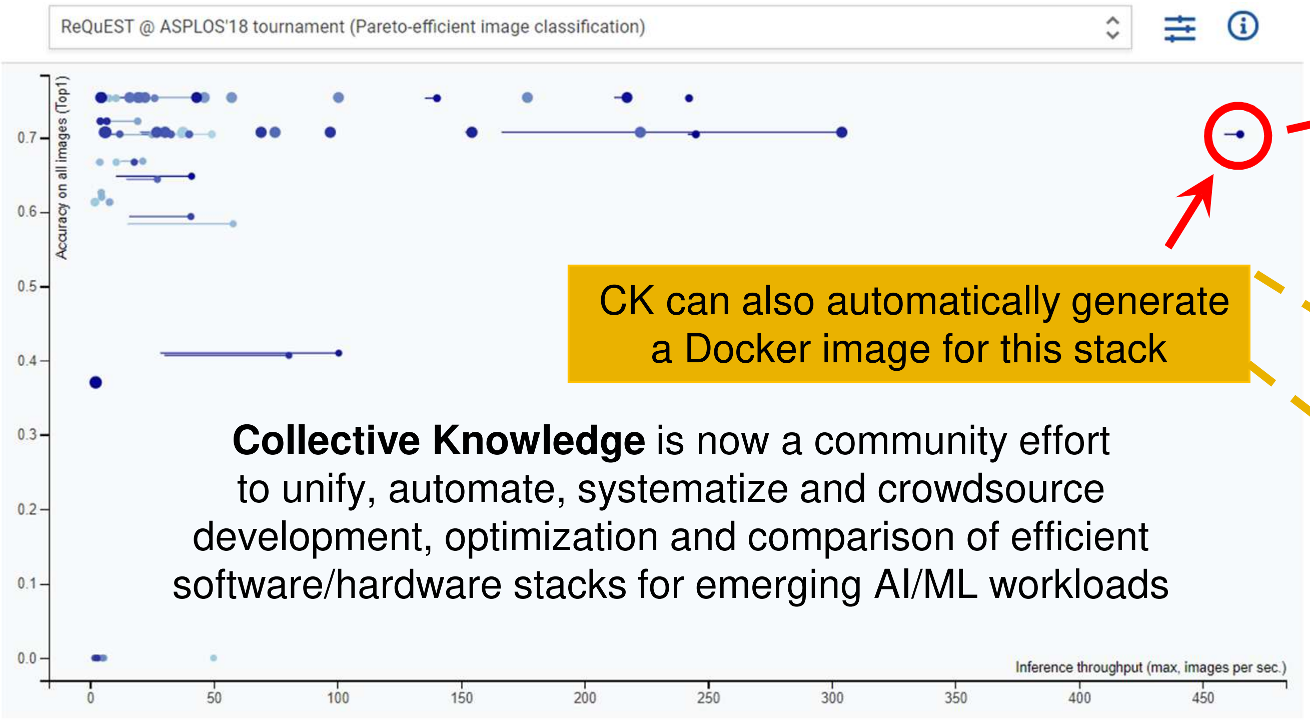

Plot dimension X: Inference throughput (max, images per sec.)

We are not announcing a single winner!

We show all multi-dimensional results at cKnowledge.org/repo and let end-users select best ML/SW/HW stacks depending on their multiple constraints!

Other companies managed to quickly reproduce results and started using CK

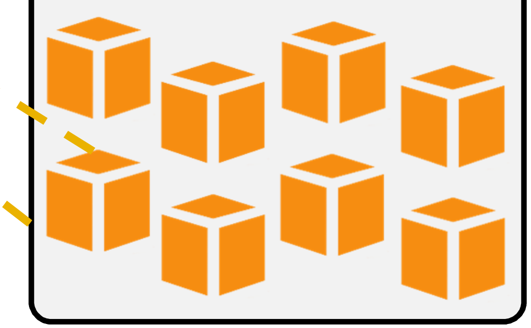
Multi-objective results for all AI/SW/HW stacks are presented on a live scoreboard
and become available for public comparison and further customization, optimization and reuse!



CK can also automatically generate a Docker image for this stack

Collective Knowledge is now a community effort
to unify, automate, systematize and crowdsource
development, optimization and comparison of efficient
software/hardware stacks for emerging AI/ML workloads

CK assists
AWS market place
with collaboratively
optimized AI/ML stacks



amazon
webservices

Accelerate technology transfer: companies can now quickly validate published techniques in their production environment using shared CK workflows (**days instead of months**)!

See joint Amazon-dividiti presentation at O'Reilly AI conference (October 2018):

conferences.oreilly.com/artificial-intelligence/ai-eu/public/schedule/detail/71549

Dividiti shared CK workflows to autotune MobileNets designs

“**MobileNets**: Efficient Convolutional Neural Networks for Mobile Vision Applications” (Andrew G. Howard et al., 2017, https://arxiv.org/abs/1704.04861):

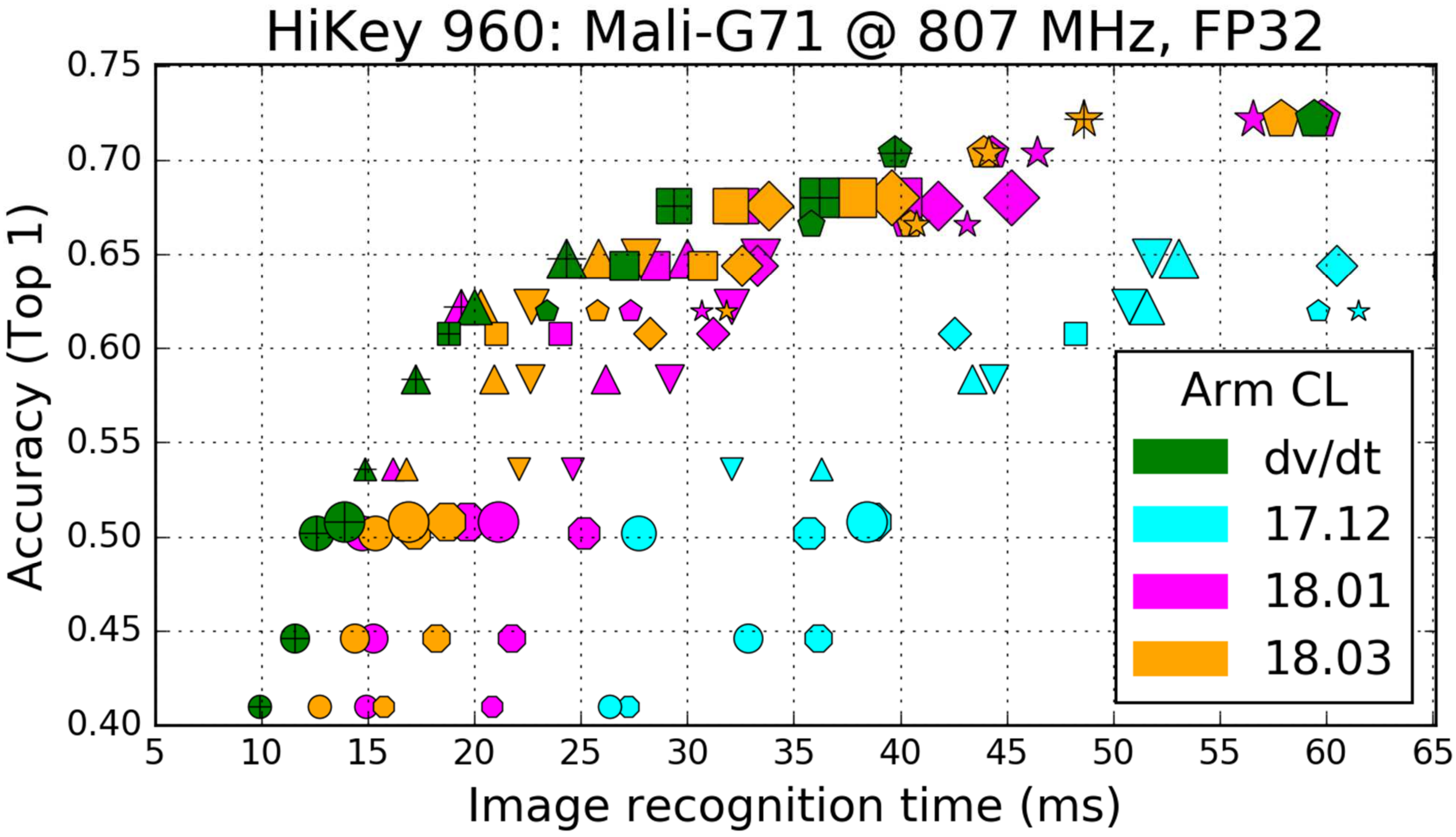
- Parameterised CNN family using depthwise separable convolutions.
- Channel multiplier: 1.00, 0.75, 0.50, 0.25 - marker shape (see below).
- Input image resolution: 224, 192, 160, 128 - marker size.

Arm Compute Library: open-source, optimised for Neon CPUs and Mali GPUs.

- 2 convolution approaches - marker shape depends on channel multiplier:
  - “Direct”: 1.00 - pentagon, 0.75 - square, 0.50 - triangle-up, 0.25 - circle.
  - “Matrix-multiplication” (MM):
    1.00 - star, 0.75 - diamond, 0.50 - triangle-down, 0.25 - octagon.
- 4 library versions - marker colour:
  - “17.12”: no opts; “18.01”: dividiti’s direct+MM opts;
    “18.03”: Arm’s MM opts; “dv/dt”: dividiti’s new direct opts.

https://github.com/dividiti/ck-request-asplos18-mobilenets-armcl-opencl

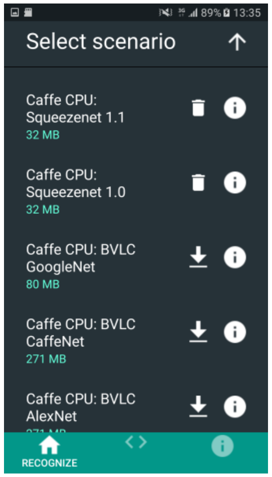
Public results autotuning MobileNets designs using Arm Compute Library



https://github.com/dividiti/ck-request-asplos18-mobilenets-armcl-opencl

CK can crowdsource experiments across Android devices provided by volunteers

Continuously collect statistics, bugs and misclassifications at cKnowledge.org/repo



Select scenario

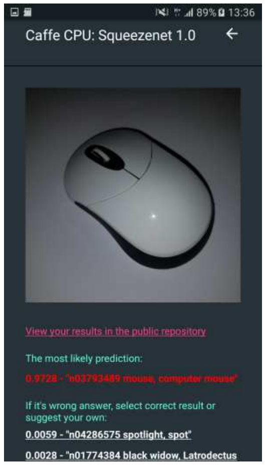
Caffe CPU: Squeezenet 1.1
32 MB

Caffe CPU: Squeezenet 1.0
32 MB

Caffe CPU: BVLC GoogleNet
80 MB

Caffe CPU: BVLC CaffeNet
271 MB

Caffe CPU: BVLC AlexNet



Caffe CPU: Squeezenet 1.0

View your results in the public repository

The most likely prediction:

0.9728 - "n03793489 mouse, computer mouse"

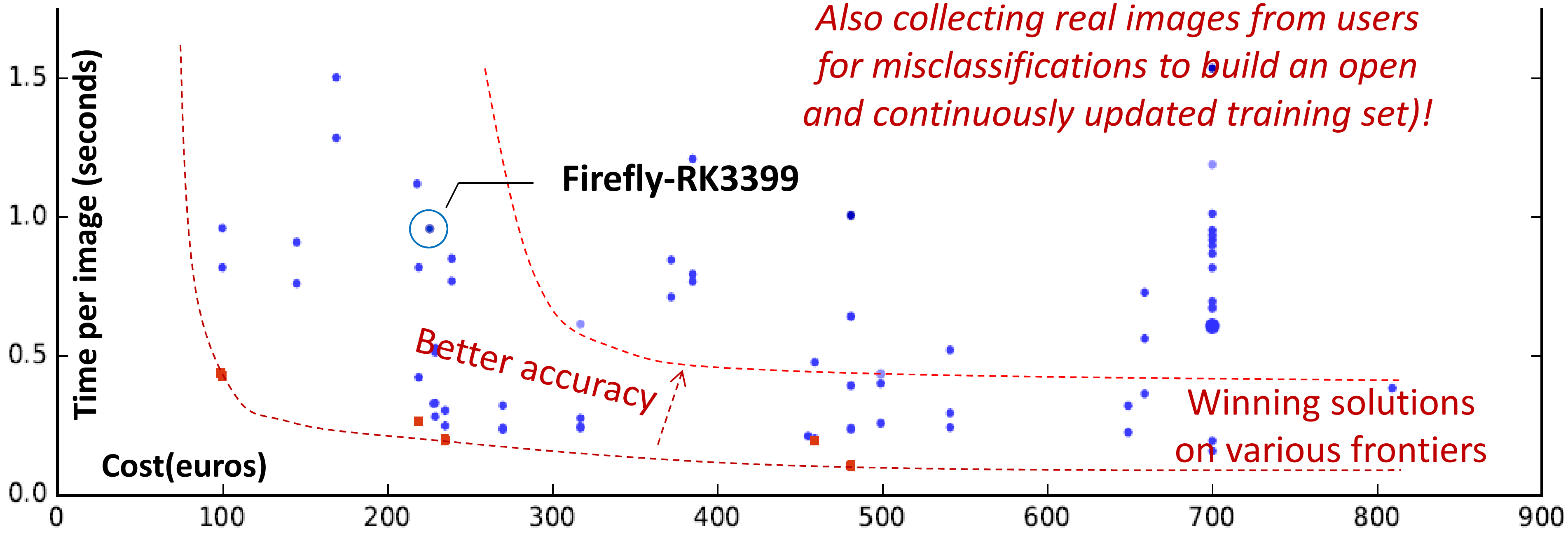
If it's wrong answer, select correct result or suggest your own:

0.0059 - "n04286575 spotlight, spot"

0.0028 - "n01774384 black widow, Latrodectus

The number of distinct participated platforms:**800+**
The number of distinct CPUs: **260+**
The number of distinct GPUs: **110+**
The number of distinct OS: **280+**
Power range: **1-10W**

No need for a dedicated and expensive cloud – volunteers help us validate research ideas similar to SETI@HOME



Also collecting real images from users for misclassifications to build an open and continuously updated training set)!

cknowledge.org/dnn-crowd-benchmarking-results

Let's dig further – (crowdsource) BLAS autotuning in Caffe on Firefly-RK3399

Tunable parameters of OpenCL-based BLAS (**github.com/CNugteren/CLBlast**)
For now only two data sets (small & large)

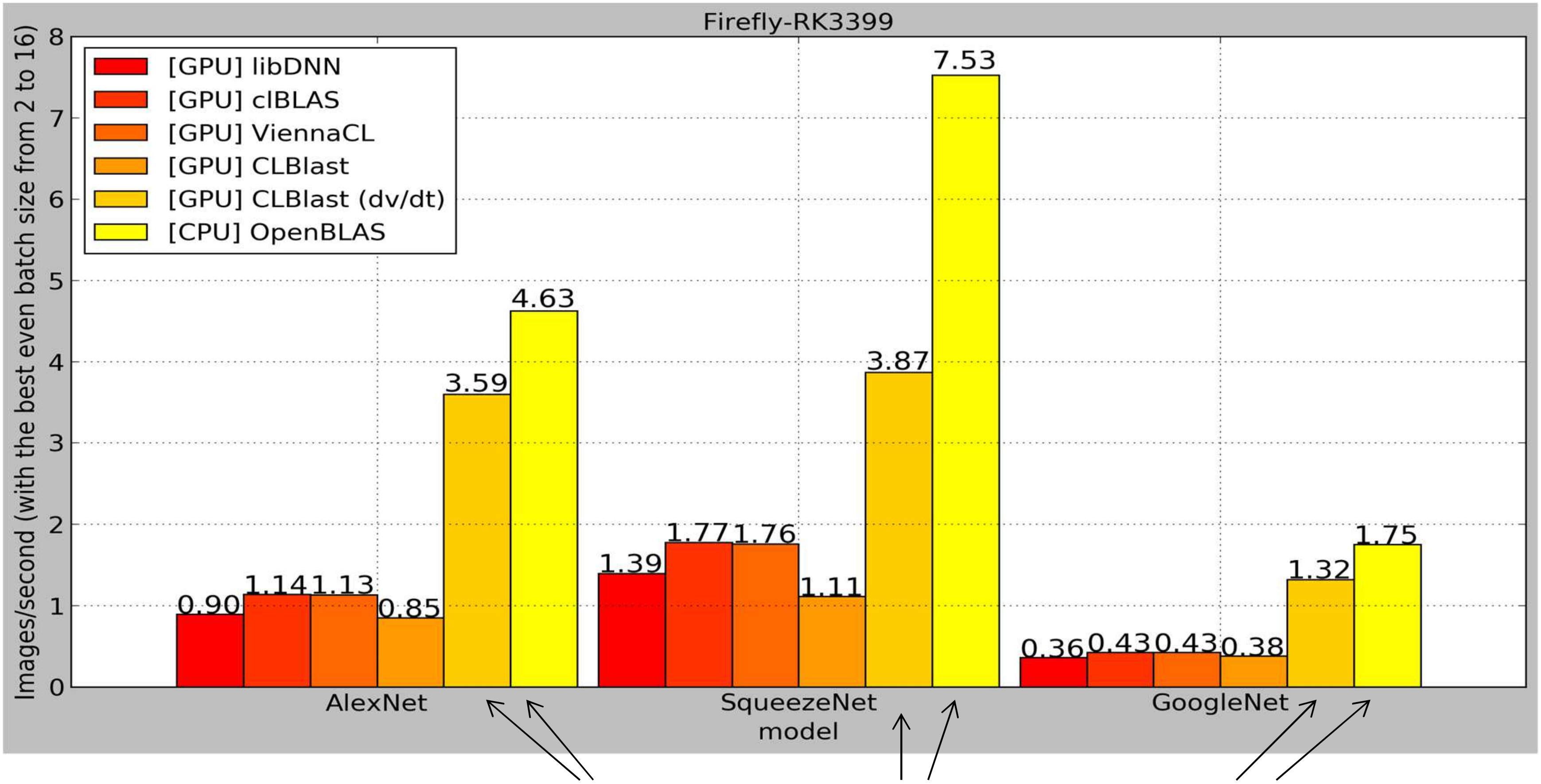
Name	Description	Ranges
KWG	2D tiling at workgroup level	{32,64}
KWI	KWG kernel-loop can be unrolled by a factor KWI	{1}
MDIMA	Local Memory Re-shape	{4,8}
MDIMC	Local Memory Re-shape	{8, 16, 32}
MWG	2D tiling at workgroup level	{32, 64, 128}
NDIMB	Local Memory Re-shape	{8, 16, 32}
NDIMC	Local Memory Re-shape	{8, 16, 32}
NWG	2D tiling at workgroup level	{16, 32}
SA	manual caching using the local memory	{0, 1}
SB	manual caching using the local memory	{0, 1}
STRM	Striding within single thread for matrix A and C	{0,1}
STRN	Striding within single thread for matrix B	{0,1}
VWM	Vector width for loading A and C	{8,16}
VWN	Vector width for loading B	{0,1}

Some extra constraints
to avoid illegal
combinations

Use different autotuners
and ML to speed up
design space exploration
based on probabilistic
focused search,
generic algorithms,
deep learning, SVM, KNN,
MARS, decision trees …

Collaboration between Marco Cianfriglia (Roma Tre University), Cedric Nugteren (TomTom), Flavio Vella, Anton Lokhmotov and Grigori Fursin (dividiti)

Let's dig further – autotuning BLAS (CLBlast) in Caffe on Firefly-RK3399

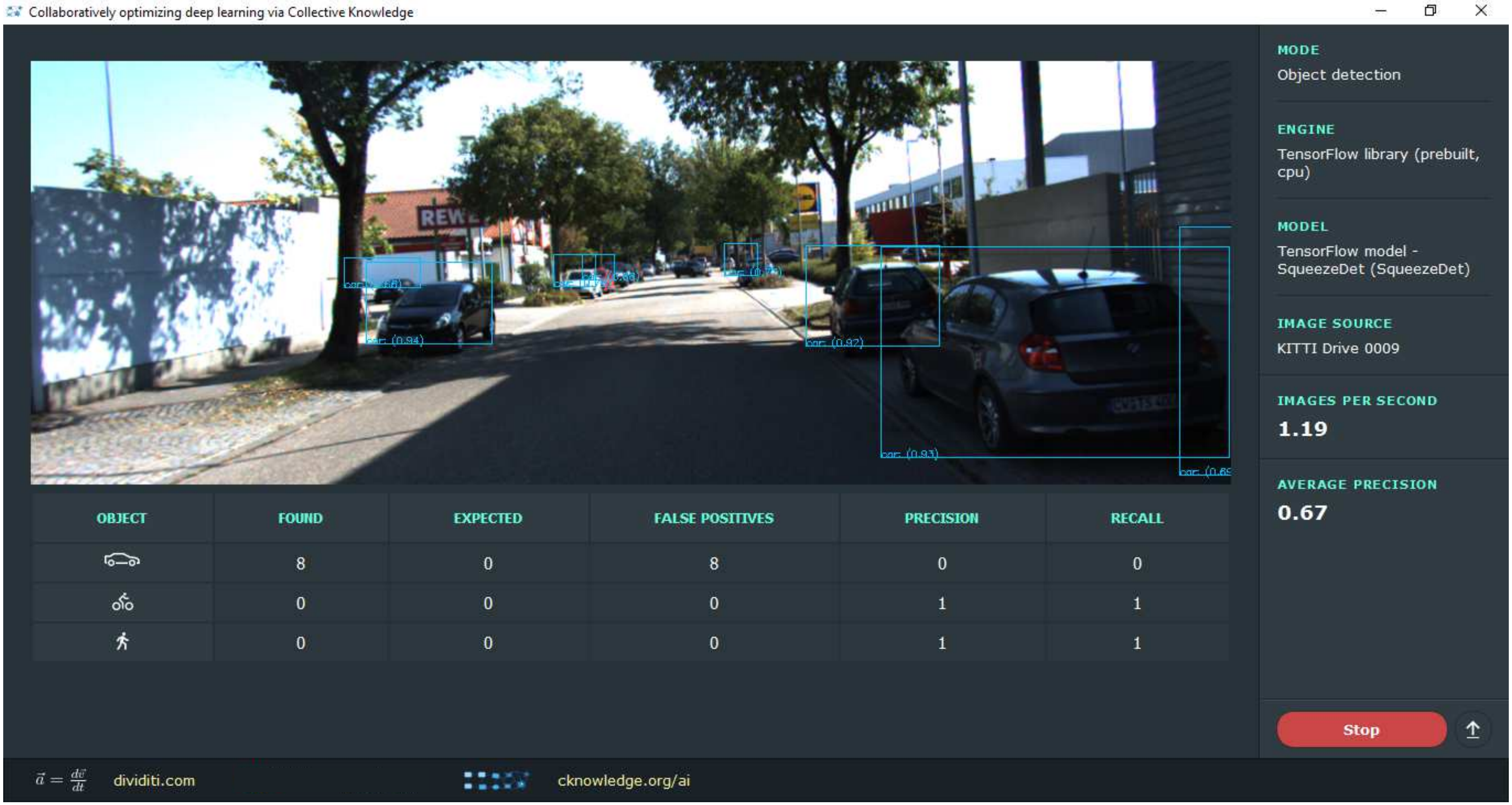


- Caffe with autotuned OpenBLAS (threads and batches) is the fastest
- Caffe with autotuned CLBlast is 6..7x faster than default version and competitive with OpenBLAS-based version– now worth making adaptive selection at run-time.

Sharing results in a reproducible way with the community for validation and improvement:

https://nbviewer.jupyter.org/github/dividiti/ck-caffe-firefly-rk3399/blob/master/script/batch_size-libs-models/analysis.20170531.ipynb

General Motors uses CK to select the most efficient platforms



Collaboratively optimizing deep learning via Collective Knowledge

OBJECT	FOUND	EXPECTED	FALSE POSITIVES	PRECISION	RECALL
🚗	8	0	8	0	0
🚲	0	0	0	1	1
🚶	0	0	0	1	1

MODE
Object detection

ENGINE
TensorFlow library (prebuilt, cpu)

MODEL
TensorFlow model - SqueezeDet (SqueezeDet)

IMAGE SOURCE
KITTI Drive 0009

IMAGES PER SECOND
1.19

AVERAGE PRECISION
0.67

CK workflows to evaluate DNN for live object detection and classification across Nvidia, AMD, ARM and Intel platforms (CUDA, OpenCL, OpenMP …)

Performance, accuracy, power consumption practically never match official reports!

GM presentation about using CK: www.youtube.com/watch?v=1ldgVZ64hEI

CK is used to collaboratively advance quantum computing (QCK)

cKnowledge.org/quantum

Quantum computers have the potential to solve certain problems dramatically faster than conventional computers, with applications in areas such as machine learning, drug discovery, materials, optimization, finance and cryptography.

We are building Quantum Collective Knowledge (QCK)
to help researchers share, compare or optimize different algorithms
across conventional and quantum platforms

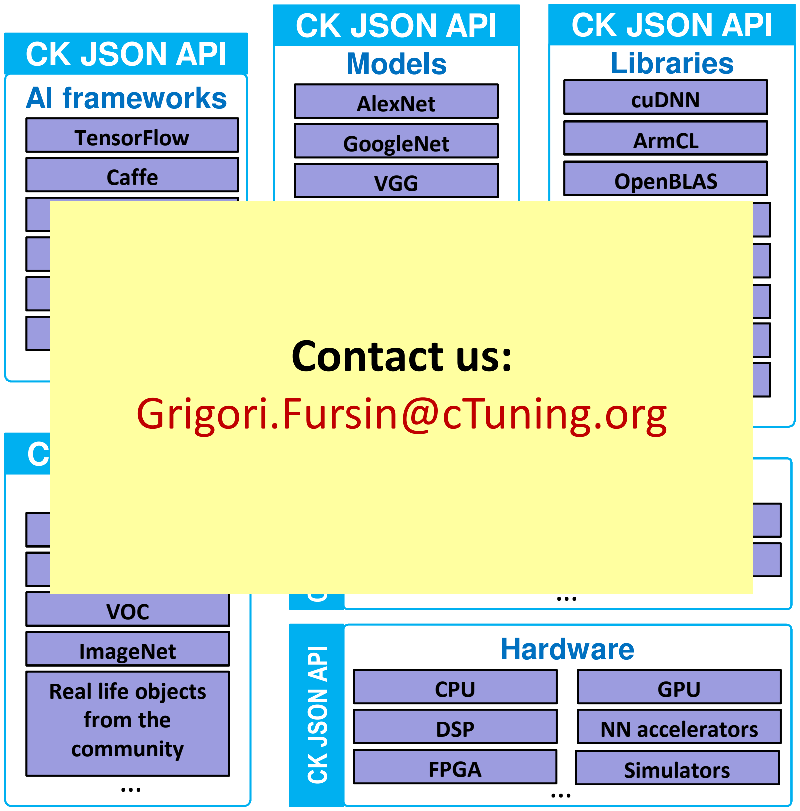
QCK helped us to organize the 1st quantum hackathon in Western Europe with Rigetti and share all results, workflows and components for further reuse

cKnowledge.org/repo

For example, we improved these workflows to support the 2nd quantum hackathon and compare results on a real 16-bit quantum machine provided by IBM!

Interested to join/support our community effort to reboot & accelerate open science?

Repositories of customizable, portable and reusable research components



CK JSON API
AI frameworks
TensorFlow
Caffe

CK JSON API
Models
AlexNet
GoogleNet
VGG

CK JSON API
Libraries
cuDNN
ArmCL
OpenBLAS

C

VOC
ImageNet
Real life objects from the community
...

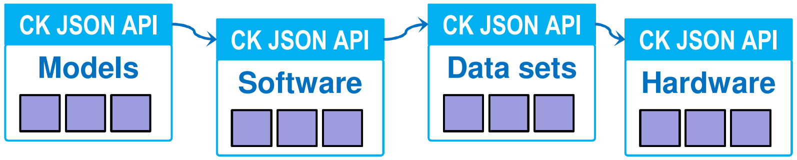
...

CK JSON API
Hardware
CPU
GPU
DSP
NN accelerators
FPGA
Simulators
...

Contact us:
Grigori.Fursin@cTuning.org

Customizable CK workflows
for real-world user tasks

Assemble scenarios such as image classification as LEGO™



CK JSON API
Models

CK JSON API
Software

CK JSON API
Data sets

CK JSON API
Hardware

Share complete workflows along with published papers
to automate artifact evaluation
and help the community build upon prior work

Crowdsource experiments with the help of volunteers
across diverse models, data sets and platforms

Present best results, workflows and components
on a live scoreboard for fair comparison and reuse
cKnowledge.org/repo

Help students learn multidisciplinary techniques,
quickly prototype new ones,
validate them in practice with companies,
and even contribute back new research components

Help companies select the most appropriate workflows,
reduce R&D costs, accelerate adoption of new techniques!