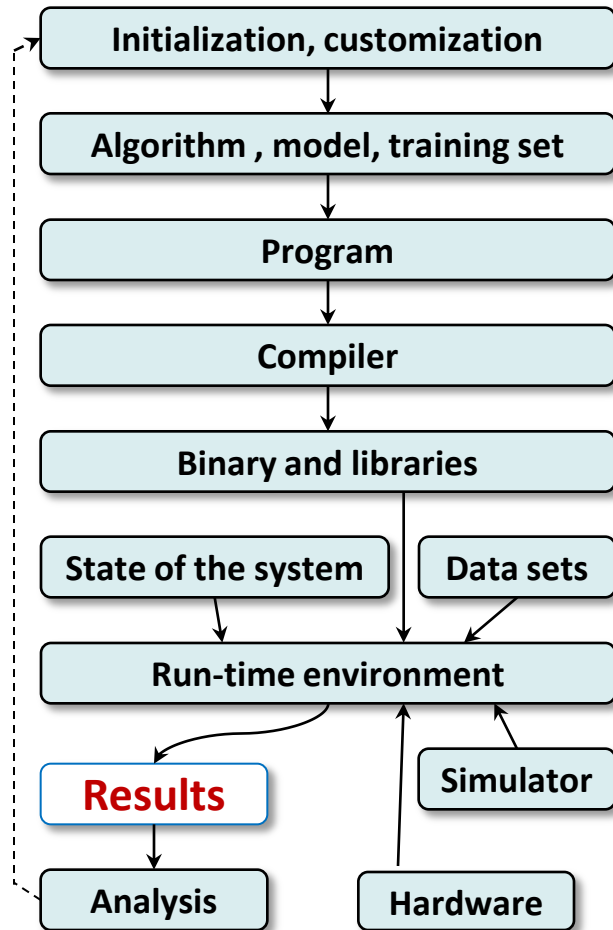


## Typical directory structure of an ad-hoc experimental pack shared for Artifact Evaluation

### Experiment workflow



(a)

**scripts/** download\_dataset.sh  
 download\_dnn\_model.bat  
 init\_setup\_rpi3\_gcc7.1.0.sh  
 init\_setup\_windows10.bat  
 init\_setup\_android.sh  
 list\_programs.sh  
 compile\_program.py  
 run\_program.py  
 analyze\_results.sh  
 build\_predictive\_model.bat  
 plot\_graph.sh

**programs/** bzip2  
 classify-image  
 decode-video-stream

**datasets/** jpg-images/ 1.jpg, 2.jpg, 3.jpg  
 png-images  
 videos

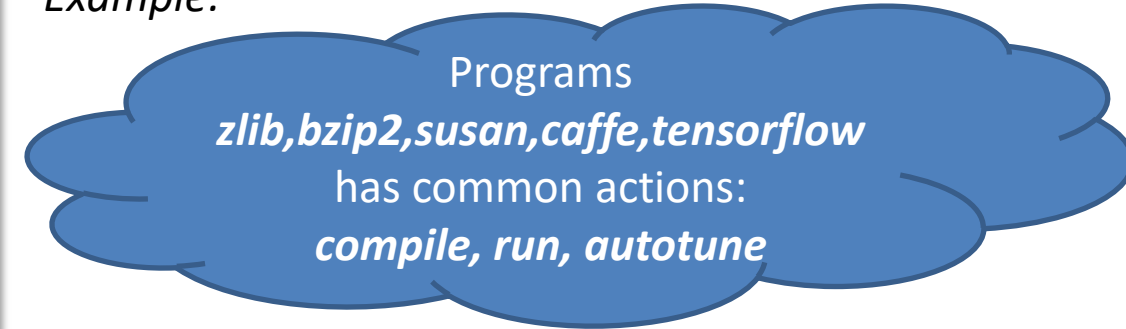
**third-party-tools/** gcc-7.1.0  
 llvm-4.0  
 openc1-profiler, cuda-profiler  
 arch-simulator  
 caffe, caffe2, tensorflow, cntk,  
 mxnet, clblast, openblas, libdnn

**some-meta/** gcc-7.1.0-compiler-flags.txt  
 llvm-4.0-compiler-flags.txt  
 rpi3-hw-description.txt

**some-results/** reference-speedups.txt  
 predictions.cvs  
 graph-autotuning-rpi3.xls

Reorganize objects with common actions and meta info into 2 level directories with auto-generated Unique ID, Python module (object API) and JSON meta information

Example:



## Convert code and data into CK repo (2 main dir. levels)

1 level (API)	2 level (entry)	
<b>.ckr.json</b>		- repo desc. including deps on other repos
<b>module</b>	<b>/program</b>	/module.py – unified CK JSON API (functions: compile, run, autotune)
<i>must be the same</i>		<b>/.cm/meta.json</b> – JSON description of a program module
<b>program</b>	<b>/zlib</b>	/*.c ... - program sources, build files
	<b>/.cm</b>	<b>/.cm/meta.json</b> - JSON desc of zlib
<b>.cm</b>	<b>/*</b>	<b>/*</b> - UID for zlib
		- UIDs for module and program

(b)