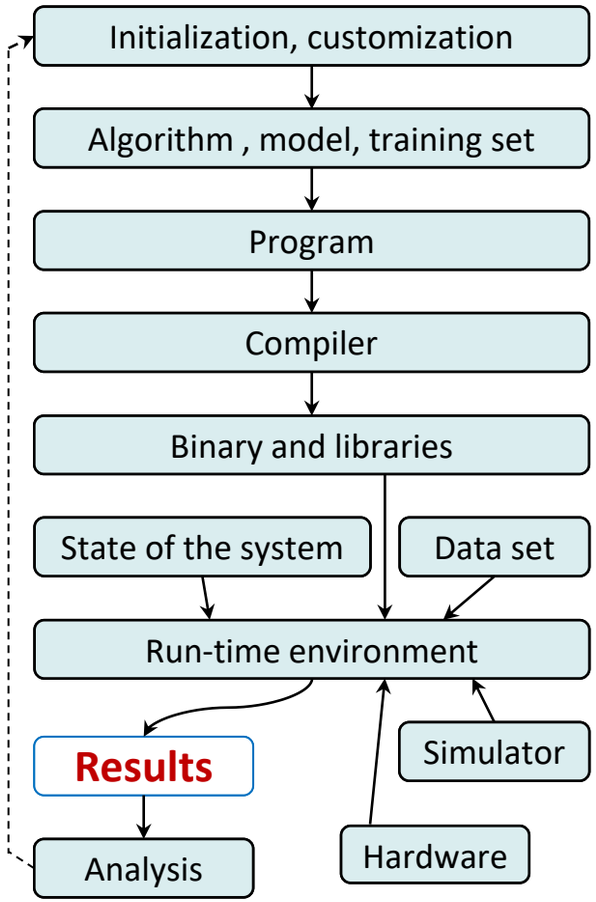
Typical directory structure of an ad-hoc experimental pack shared for Artifact Evaluation



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
opencl-profiler , cuda-profiler
arch-simulator
caffe, caffe2, tensorflow, cntk, mxnet, clbast, 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

Organize objects with common actions and meta info into discoverable, reusable and shareable Collective Knowledge entries with auto-generated Unique ID, JSON meta data and Python module (wrapper) with unified JSON API

Example:

Programs
zlib,bzip2,susan,caffe,tensorflow
has common actions:
compile, run, autotune

Pack them to a CK repository (2 CK dir levels)

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