Dockerfile in-depth docs (AIDAmri v1.1.1)

AIDAmri/Dockerfile at https://github.com/aswendtlab/AIDAmri/tree/dockerdev
Latest commit: e9f2f5e2943fb5d00924da345a0791c857e00f03
Latest file change: 2022/08/25
Latest doc change: 2022/09/27

Please make yourself comfortable with Dockerfile synthax first.
The layers will be referred to their order of appearance within the file.
Passages highlighted in red refer to tools that were not individually tested, their usage is assumed within the different installation processes, or issues that were not yet addressed exhaustively.

LAYER 1 (**FROM**); l.1

- Loads Ubuntu 18.04 base image
  - Ubuntu 18.04 might be deprecated. 20.04 or 22.04 might be more feasible for applications like DSI-Studio
  - however, DO NOT CHANGE the Ubuntu version, unless you commit to fix the upcoming issues (e.g. FSL need to be implemented differently)

LAYER 2 (**ENV**); l.3

- “DEBIAN_FRONTEND=noninteractive” will automatize dialogue that appears when installing packages as the building process will not handle dialogue itself. Every requested answer will be set to default.

LAYER 3 (**RUN**); l.5-16

- apt-get update/upgrade/install -y (ll.6f)
  - apt-get (or apt) is an package managing tool
    - the update option resynchronizes the package index files to its sources
    - the upgrade option upgrades every package to its most recent version
    - the -y flag in both cases sets the dialogue option to “yes” to execute the directives
    - this is a commonly standard procedure before installation within Linux command line for managing packages
    - install is self-explanatory. All following packages (ll.8-16) are queued for installation
- installing packages
  - wget: downloading tool for FTP/HTTP(S) server data; needed for DSI-Studio
  - unzip: tool for ZIP-file extraction; needed for DSI-Studio
  - build-essential, checkinstall, zlib1g-dev:
    - Different tools for debian package building and decompression
    - Assumed usage within FSL installation
      - Exact usage unknown as it was transferred from similar Docker building processes including cmake
  - libssl-dev: Secure Socket Layer tool package for crypto protocols (basically for save internet protocol usage)
  - git: tool for downloading DSI github repository
  - dc: arbitrary precision calculator; usage not exactly known
  - ffmpeg: Library for storing audio and video; not sure if really needed but might be in use for NiftyReg data management or it is an artefact of the DSI-Studio GUI usage
  - libsm6: Session management library; usage unknown

    - libxext6: Misc. extension library; usage unknown

LAYER 4 (**RUN**); ll.18-24
- This part sets up cmake to build and compile NiftyReg within the image
- It gets downloaded via wget (l.18), extracted via tar (l.19) and installed via the bootstrap script (l.22) and build by make and make install (ll.23f)
    - Note that the tar archive is removed after extraction (l.20)
    - The tar extraction uses the flags -x for “extract”, -v for “verbose” (so the extraction process is echoed; it is however optional), -z to handle compressed data and -f to pass the archive name to the command

LAYER 5 (**RUN**); l.27
- mkdir aida creates a directory that will contain binaries and libraries from AIDAmri and function as the working directory within a running container

LAYER 6 (**WORKDIR**);l.28
- Sets working directory (might be a not needed as this step is done later again; CHECK FOR DEPRECATION)

LAYER 7 (**RUN**); ll.31-33
- installing python3, pip and venv (virtual environment)

LAYERS 8-10 (**ENV, RUN, ENV**); ll.34-36
- Setting up Python environment path

LAYER 11 (**RUN**); l.37
- Upgrading setuptools (used for python package builds in next layer)

LAYER 12 (**COPY**); l.38
- Copies the Python requirements list into image (requirements.txt). It is important that this file is in the same folder as the Dockerfile

LAYER 13 (**RUN**); ll.39f
- installs all required packages via pip

LAYER 14 (**COPY**); l.44
- Copies a modified FSL installation script. The modification is done so a dialogue during the installation can be avoided. Non-interactive mode does not solve the issue. Piping the answer instead of modifying the file might work as well

LAYER 15 (**RUN**); l.45
- Installation of FSL. The version 5.0.11 is apparently required for AIDAmri, however other versions were not tested yet

LAYERS 16-20 (**ENV, RUN, ENV, ENV, RUN**); ll.47-52
- Those layers set up the environment variables for FSL
- l.48: The FSL directory is addressed
- l.49: Running the fsl bash file for set-up
- l.50: The output type for FSL is specified (.nii.gz)
- l.51: The binary path for FSL commands is addressed
- l.52: The binary paths are exported to make them executable

LAYER 21 (**RUN**); l.55
- directory is made for NiftyReg

LAYER 22 (**WORKDIR**);l.56
- changes working directory to NiftyReg

LAYER 23 (**RUN**); ll.57-65
- NiftyReg is set up via cmake directives (piped in CLI)
- NiftyReg is cloned and set up according to AIDAmri installation manual
  - Apparently a earlier commit was used (l.59 hard reset to specific commit). It is unknown to which extend it matters to use earlier versions of NiftyReg. This has yet to be tested, however it does not affect the functionality of the pipeline
- cmake directives (ll.61ff) set the build types, installation prefix and the C-compiler (used compiler: GCC v7)
  - NiftyReg installation currently throws a lot of warnings while building image. They do not seem to affect the program at back-end usage but might inflict hidden issues. To fix them it might be recommended to adjust the cmake settings for proper compilation

LAYER 24 (**RUN**); l.66
- Exporting the installation environmental variable to the proper directory

LAYER 25 (**ENV**); l.67
- NiftyReg binary path is settings

LAYER 26 (**ENV**); l.68
- LD library paths is set for the program to search in shared libraries

LAYER 27 (**ENV**); l.69
- exporting paths

LAYER 28 (**WORKDIR**);l.71
- Changing working directory to aida

LAYER 29 (**RUN**); l.73-75
- Downloads and unzips recent Ubuntu 18 version of DSI-studio archive
- rm in l.75 to remove the zip file after extracting

LAYERS 30,31 (**COPY**, **COPY**); ll.78f
- Copying the binary and library folder from AIDAmri into the image

LAYER 32 (**RUN**); ll.80ff
- Paths for dsi-studio and the custom atlas are written into the dsi-studio-path text file for calling DSI via DTI connectivity protocols
  - May be obsolete if DSI is changed to a compiled version within the image instead of calling the binary
  - Custom atlas directories not currently in use