Tensorflow and Keras installation steps for Deep Learning applications in Rstudio

Ricardo Dalagnol[1] and Fabien Wagner

1. Install OSGEO (https://trac.osgeo.org/osgeo4w/) to have GDAL utilities and QGIS. Do not modify the installation path.
2. Install ImageMagick (https://imagemagick.org/index.php) for image visualization.
3. Install the latest Miniconda (https://docs.conda.io/en/latest/miniconda.html)
4. Install Visual Studio
    a. Install the 2019 or 2017 free version. The free version is called 'Community'.
    b. During installation, select the box to include C++ development
5. Install the latest Nvidia driver for your GPU card from the official Nvidia site
6. Install CUDA NVIDIA
    a. I installed CUDA Toolkit 10.1 update2, to check if the installed or available NVIDIA driver is compatible with the CUDA version (check this in the cuDNN page https://docs.nvidia.com/deeplearning/sdk/cudnn-support-matrix/index.html)
    b. Downloaded from NVIDIA website, searched for CUDA NVIDIA download in google: https://developer.nvidia.com/cuda-10.1-download-archive-update2 (list of all archives here https://developer.nvidia.com/cuda-toolkit-archive)
7. Install cuDNN (deep neural network library) – I have installed cudnn-10.1-windows10-x64-v7.6.5.32 for CUDA 10.1, the last version https://docs.nvidia.com/deeplearning/sdk/cudnn-install/#install-windows
    a. To download you have to create/login an account with NVIDIA Developer
    b. Before installing, check the NVIDIA driver version if it is compatible
    c. To install on windows, follow the instructions at section 3.3 https://docs.nvidia.com/deeplearning/sdk/cudnn-install/#install-windows
    d. To check for env vars Win+R "control sysdm.cpl"
8. Install R v4.X (https://cran.r-project.org/bin/windows/base/)
9. Install Rstudio (http://www.rstudio.com/products/rstudio/download/)
10. Install Rtools (https://cran.r-project.org/bin/windows/Rtools/) and add in the path. This is explained in the link above how to do this from R. It is just a line of code.
11. Create and setup a virtual environment with conda prompt and install python 3.7, tensorflow-gpu and keras. To do this open Anaconda Prompt in Start menu, then type in the commands in the Table 1, one by one. If a prompt appears asking y or n, type y.

Table 1 – Commands to create and setup the virtual environment.

Command	**Reason**
conda create -n r-tensorflow python=3.7 anaconda	Create an environment named r-tensorflow inside conda with the python v3.7
activate r-tensorflow *Obs: depending on the conda version you may need to type conda activate r-tensorflow*	Activate the conda environment so we can start installing stuff

[1] Contacts : ricds@hotmail.com and wagner.h.fabien@gmail.com
Send an email for collaboration or if you have suggestions for the improvement of this manual.

pip install tensorflow-gpu==2.2.0-rc4	Install the tensorflow v2.2
conda install pillow	Need this to work with .tif
conda install libtiff=4.0.10	Need this to work with .tif
pip install tensorflow-addons	Install tensorflow addons
python	Start python so we can quickly test if both python and tensorflow are working
import tensorflow as tf	Try to import tensorflow. If no error, you are fine; Else, start the installation again and/or try to find the source of error
import numpy	Try to import numpy; another important library. If no error, you are fine; Else, start the installation again and/or try to find the source of error
exit()	Exit python
deactivate *Obs: depending on the conda version you may need to type conda deactivate*	Deactivate the conda environment

If all worked, the environment should be ready to use in Rstudio.

12. Now install the packages in R. Open Rstudio and run the commands below. If it asks if you want to install from source, click yes.

```
install.packages("devtools")
library(devtools)
devtools::install_github("rstudio/keras")
install.packages("tensorflow")
install.packages("reticulate")
library(keras)
library(tensorflow)
library(reticulate)
```

13. If everything installed correctly, it is time for testing training a model.
    a. Download the UNET model and data from the paper of Wagner et al. (2020) and run the code (https://zenodo.org/record/3926822#.Xv4c1ChKiUl).
    b. If everything is working properly, you should see the information regarding UNET training being outputted in the console (Figure 1) and the variation in training and validation loss and accuracy metric in the viewer (Figure 2). In Figure 2, see that the training (blue) goes up quite fast at the beginning and then slowly increases until the end of the training. Meanwhile, the validation (green) only increases after 25 epochs. Your outputs will not be exactly identical to this, but this is expected.

14. The end. Have fun with deep learning!

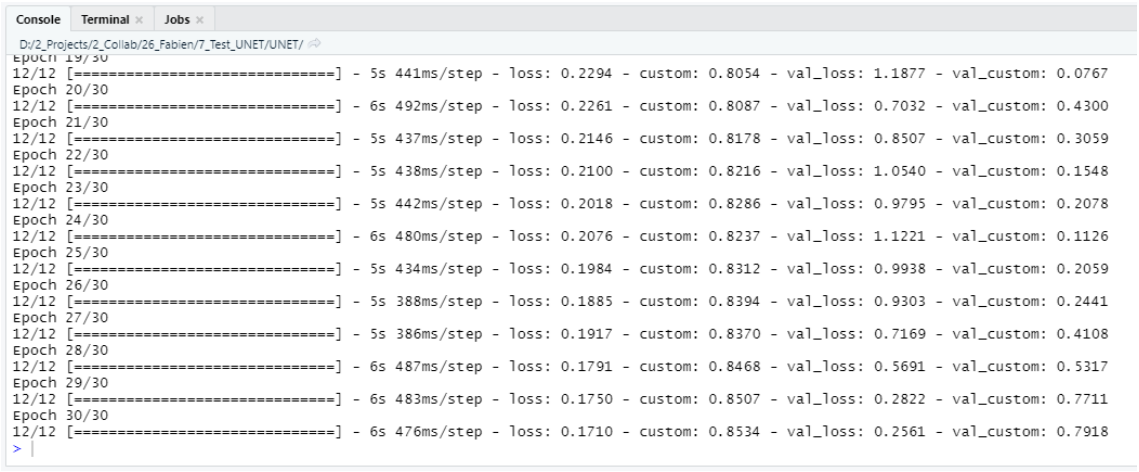

Figure 1 – Information output in the console during training.

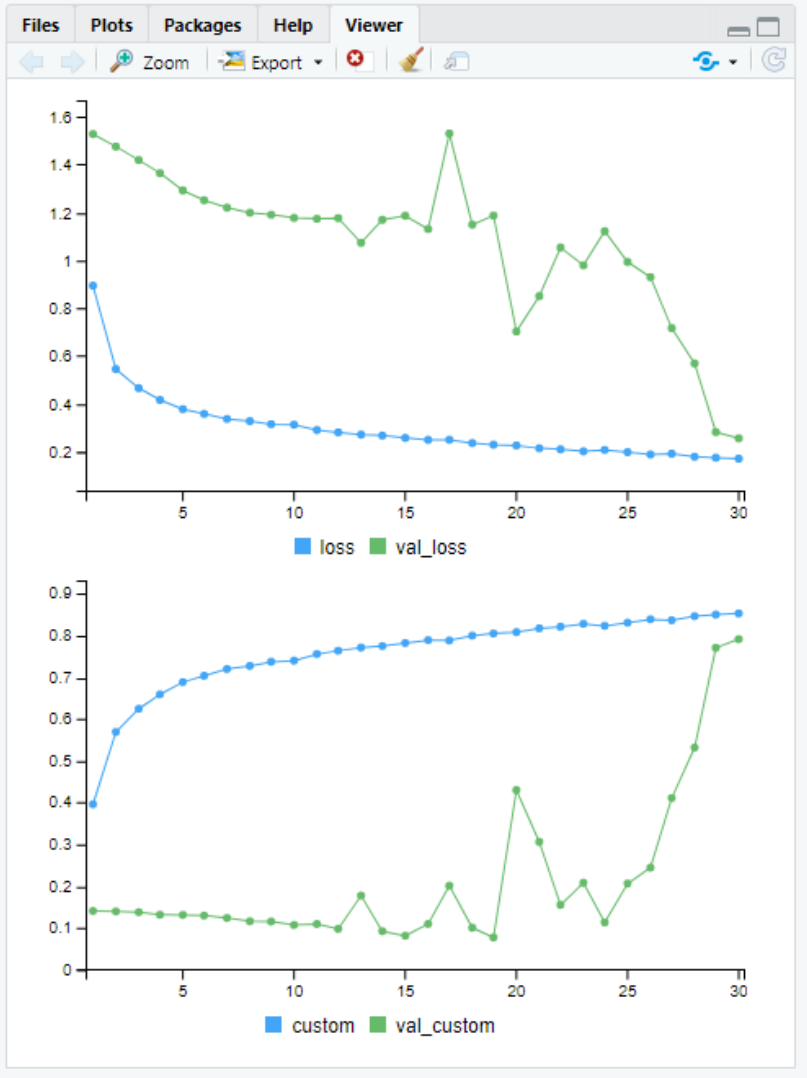

Figure 2 – Graph of training and validation loss (up) and accuracy metric (bottom).