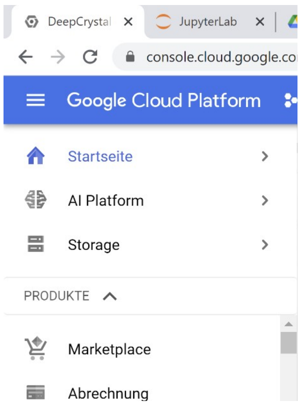
User Manual «Training and Predicting Crystal Images Using InceptionV3»

Introduction

This short User Manual guides you through the steps of using the Jupyter notebooks created during the final project “Crystal Detection” in collaboration with Propulsion Academy in Zürich from 26th of October to the 20th of November 2020. The goal was to train a deep learning model which can eventually precisely detect crystals on the images provided by Crystals First GmbH. Throughout the process, four different models were tested (ResNet50, EfficientnetB7, VGG19 and InceptionV3). Inception was finally selected because it provided the best results on the test set.

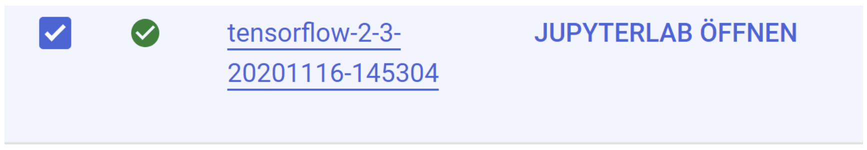
Accessing the VM

Assuming that you were granted full access to the Google Cloud Project «Deep Crystal» navigate to the menu «AI Platform» and further to «Notebooks».

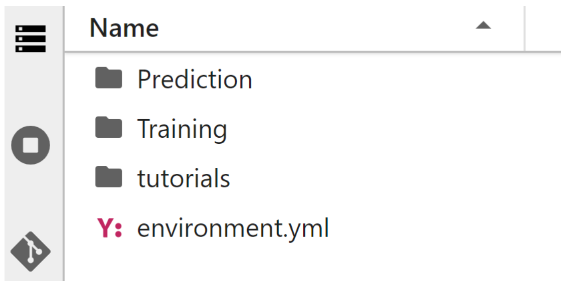


The VM instance named «tensorflow-2-3-20201126-145304» needs be running before you can open the Jupyterlab instance via klicking the button «open Jupyterlab». If the green tick is not visible, select the VM and click «Start» on the top.

When you are finished working on the VM, please make sure to **shut it down** again.



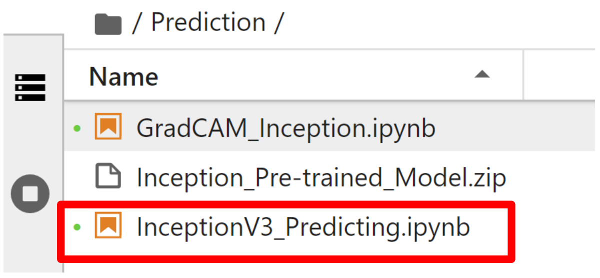
Structure of Jupyter Lab



The two folders “Prediction” and “Training” are needed to either make predicitons on the existing pre-trained model or to re-train the existing model. The environment file lists the required installations made on the VM.

Prediction only

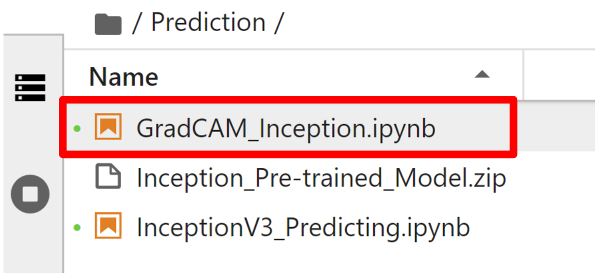
If you decide to only make predictions on an existing .tar file from the storage and use the InceptionV3 model which was created during the project with Propulsion, no previous training of the model is required. You can simply run the notebook “InceptionV3_Predicting” in the folder “Prediction”.

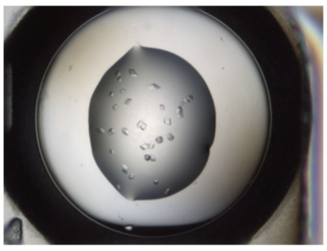


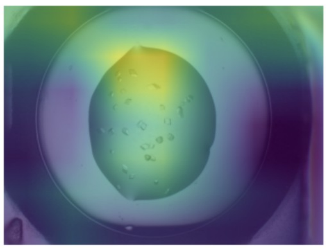
GradCAM

“GradCAM_Inception” is an optional notebook. If you are interested to see which parts of an image are important for learning, this notebook helps you understand this in more detail. For InceptionV3, this doesn’t seem to work properly, other models work better.

To be able to use this notebook you need load the pre-trained or another existing model as well as select an image which you are interested in to look at.







Re-training the Model

If you want to re-train the existing model the “Training” folder is where you need to start off.

Before you can retrain the model you need to make sure that the additional images are incorporated into the json file which is needed for the prediction of of this notebook. This includes the use of the confusion matrix, the classification report and the ROC curve.

Create a Json file

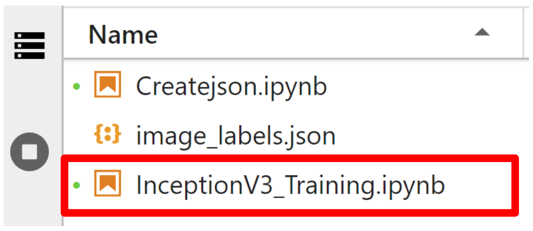
If any changes are made to the trainin_data the notebook “Createjson” needs to be run. If no changes are made to either of the folders, don’t bother changing it.

This is a possible scenario when it needs to be re-run:

You want to increase the amount of images on the training and test set:

1. Do not delete the existing Json file, use it to move the images from the test folder back to the training folder, make sure they are in the correct folder.
2. Add labeled images to the training_data, make sure the names are unique and easy (e.g. c_img_44, no_c_img_320). This simplifies the search later on.
3. Run the “Createjson notebook to create a new json file.
4. Remove 10 - 20% of the images from the training_data to the test folder.

Training


Name
Createjson.ipynb
image_labels.json
InceptionV3_Training.ipynb

In the current setting of this notebook, the previously adapted test and training folder (see Createjson) are copied from the storage to the VM temporarily. Some augmentation and balancing is done before the model InceptionV3 is loaded from the internet together with the weights. Afterwards, some fine-tuning is carried out before the actual training is started. If you want to learn more about training a model and transfer learning, there are some great tutorials from TensorFlow.

You can save the model at this point or continue with the evaluation before saving it. To save the model, make sure to give it a meaningful name or to keep a separate log where you note the different parameters for each model. Another way of keeping track of the accuracy and loss at this stage or to do some hyperparameter tuning at a very evolved stage, use Tensorboard.

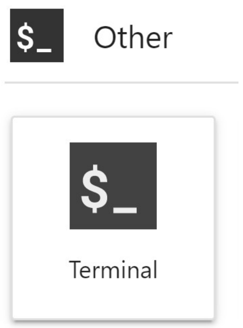
Predicting

Towards the end of this notebook there is a section called “Prediction only - loading a pre-trained model”. Here, you can load the pre-trained model and predict on new test data or you can load a newly trained model and predict on the existing test set to compare results with previous models. Which ever way you want it to work.

In order for this section to work, you need to a) have a json file that matches the current setting of you training and test folder and b) be able to load a pre-existing model.

Creating an environment file

If you want to share the requirements file with someone create an yml file from the terminal.



Launch a new the terminal and type *conda list* to see all the installations in the current VM. Then type *conda env export --from-history> environment.yml* to create the environment file. Now, manually compare the yml file with the list of installations an add the version for each requirement if it’s not specified yet.