**Software Assessment**

Write a machine learning-based algorithm that can read in a png image file and identify the size of the drill bit displayed in the image.

For training data refer to the videos posted in this folder: <https://drive.google.com/drive/folders/1-EAPMKA5vP6We_qgYzkrhMGA0X3MJAwA?usp=sharing>

The drill sizes shown in each video file are as follows:

| **Drill Size** | **Video Files** |
| --- | --- |
| 2.0 mm x 26 mm | 20x26dark.mp4, 20x26light.mp4 |
| 2.0 mm x 28 mm | 20x28dark.mp4, 20x28light.mp4 |
| 2.8 mm x 22 mm | 28x22dark.mp4, 28x22light.mp4 |
| 3.5 mm x 19 mm | 35x19dark.mp4, 35x19light.mp4 |
| 3.5 mm x 22 mm | 35x22dark.mp4, 35x22light.mp4 |
| 3.5 mm x 28 mm | 35x28dark.mp4, 35x28light.mp4 |
| 3.5 mm x 30 mm | 35x30dark.mp4, 35x30light.mp4 |
| 4.2 mm x 22 mm | 42x22dark.mp4, 42x22light.mp4 |
| 4.2 mm x 30 mm | 42x30dark.mp4, 42x30light.mp4 |

Write a program to sample frames from the provided videos to create training data for your ML pipeline. Create a model and pipeline which classifies the frames. Save your trained model.

Now create a Windows executable program (created from Python source, but a \*.exe, not a \*.py) which takes a single folder as input. The folder will contain one or more 1280x960 png images. For each image found in the folder, the program should print a line to the console which contains the filename and its classification using your trained model. At the end of the list of images, also print a table that contains each classification found, and the number of images associated with that label. The image folder name may be input on the command line or prompted for within the program. Expect that the images will be like any given frame from one of the video files and that the drill bit size will be one of the sizes listed in the table above. Expect that the size of the drill bit in the input image will match the video frames but may vary in brightness or blurriness.

In case you choose not to use Python, please use a standard development language and include a file which can be executed on Windows, such as a jar file. JavaScript that runs in a browser is fine, as well. Please do not use LabView, MATLAB or equivalent.

General Guidelines:

Your solution should be well documented. All algorithms should be documented in full, and a description of the algorithm should be provided. Discuss the strengths and weaknesses of your network, what alternative network architectures that you considered or with which you experimented, and why you think your chosen network is optimal. Describe how you tuned your model. Describe how well your model is performing using illustrations, graphs, metrics, edge cases or similar methods.

Ways to Improve your Chances:

* Good code design & structure
* An object-oriented solution
* Good documentation!

Requirements for submission

* Turn your result in by the end of the second day. Earlier if possible.
* Attach all documentation and source code in an email to **jtieman@neocis.com**.