Introduction

Core photography is a rich source of geological information that contains important textural, mineralogical and geotechnical information. Currently, core photography is underutilized in exploration and mining due to inconsistencies in the data and the arduous task of transforming historical photography into the cropped and depth-registered form required for integration with other geological data. Photographs taken on different sites are now mostly standardised, but we have millions of historic core images from 100s of thousands of drill holes. As a new generation of image analysis techniques is becoming more powerful and prevalent within exploration and mining, these large image repositories will eventually become rich sources of quantitative data. Cropping core images is an exhausting job when done manually. When the core trays aren’t in a locked position relative to the camera which is the case for much historical core images, cropping need to be manually adjusted in each image. Depending on the quality of the core image, each drill hole could take up to an hour to manually crop. If the exact extents of the core rows can be automatically defined, then this could result in up to 80 – 90% decrease in the time spent cropping drill core images manually.

Theory

The challenge is to build an algorithm that can determine and map the spatial extents of core tray rows and distinguish between different aspect ratios. We will try two different techniques in this paper.
The first one is Convolutional Neural Network. CNNs are comprised of three types of layers. These are convolutional layers, pooling layers and fully-connected layers. When these layers are stacked, a CNN architecture has been formed. The basic functionality of the CNN can be broken down into four key areas: 1) input layer, 2) convolutional layer, 3) pooling layer and 4) fully-connected layers.

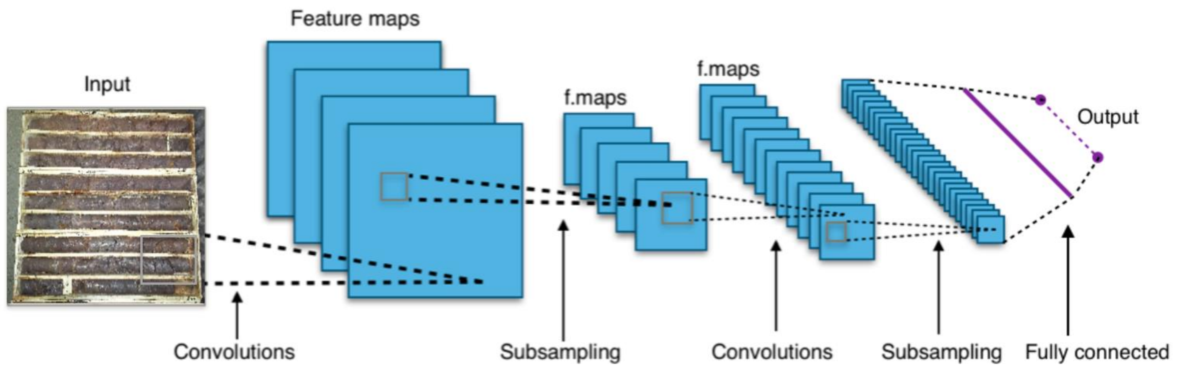

Figure 1 *An illustration of the architecture of Convolutional Neural Network (CNN).*

The second one is U-Net : Convolutional Networks for Biomedical Image Segmentation technique. The network architecture is illustrated in Figure 2. It consists of a contracting path (left side) and an expansive path (right side). The contracting path follows the typical architecture of a convolutional network. It consists of the repeated application of two 3x3 convolutions (unpadded convolutions), each followed by a rectified linear unit (ReLU) and a 2x2 max pooling operation with stride 2 for down-sampling. At each down-sampling step we double the number of feature channels. Every step in the expansive path consists of an up-sampling of the feature map followed by a 2x2 convolution (“up-convolution”) that halves the number of feature channels, a concatenation with the correspondingly cropped feature map from the contracting path, and two 3x3 convolutions, each followed by a ReLU. The cropping is necessary due to the loss of border pixels in every convolution.

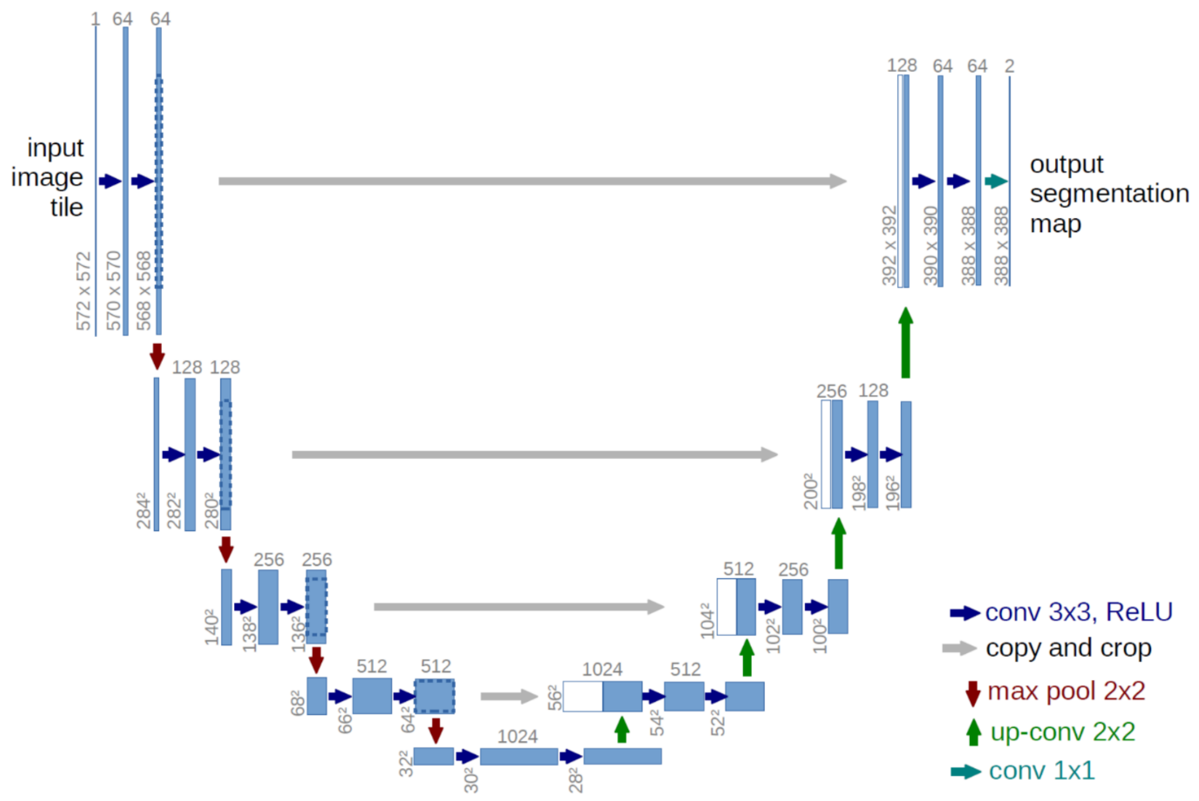

Figure 2 *U-net architecture (example for 32x32 pixels in the lowest resolution). Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations.*

The Data

The data used in this study was provided by one of the GoldSpot clients. The data is a set of images chosen at various locations chosen randomly from our drill-hole data library. The images are resized to 256 x 256 pixels and each pixel is classified as either boundary box or not boundary box. The training dataset consists of 1000 images with corresponding mask data. Of these 1000 images in the dataset, 800 images were used for training and validating the model and 200 images were held out for testing.

Pre-processing

Some adjustments to the raw dataset are made to increase pipeline speed for training and inference processes. The training, validation, and test image sets are separately concatenated into a Numpy array file and saved on disk. For images, pixel intensities are normalized between 0 and 255 to convert to 8-bit format. Brightfield image pixel values are normalized between 0 and 255 and then contrast is applied using the PIL library's ImageEnhance module in Python. This increases pixel values that are higher and decreases lower ones, hence making cellular structures stand out more. The reasoning behind it is that with the initial normalized images U-net struggled to get past a certain accuracy. After applying pre-processing, our network managed to converge to better validation accuracy.

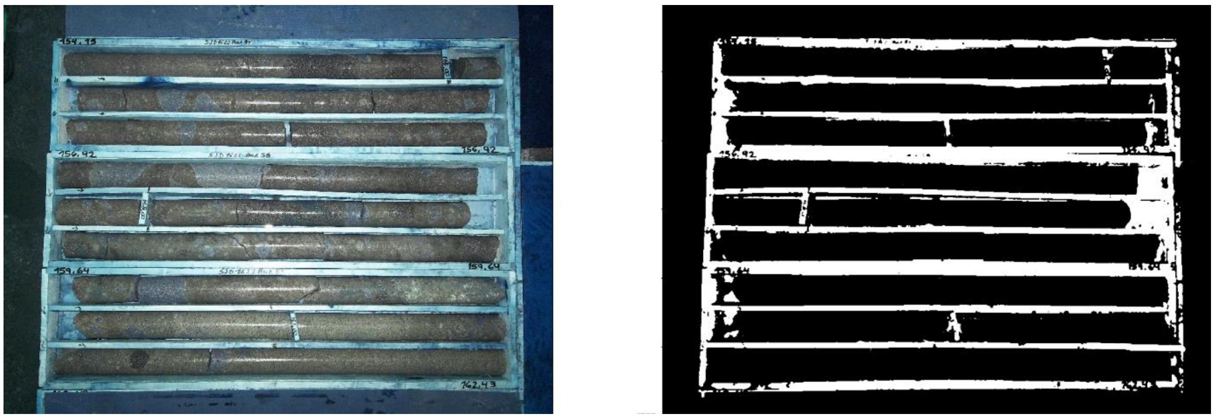
Figure 3 *Sample image and mask*

Deep Learning Libraries

Deep learning related work on the research was conducted using Keras with Tensorflow backend. Keras is an open source high-level wrapper library for neural network frameworks. Its main features include user friendliness, modularity, and ease of extensibility. Keras is written in Python which makes for an easy to understand source code. It is one of the most used high-level wrappers for Tensorflow. Developing neural networks is quicker when using it for many commonly used layer declarations. Keras is flexible with regards to using different backend frameworks for carrying out the computation-heavy tasks. Tensorflow is meant for high-performance numerical computations across various computing hardware. To achieve its performance levels, it uses optimized code for specific hardware platforms.

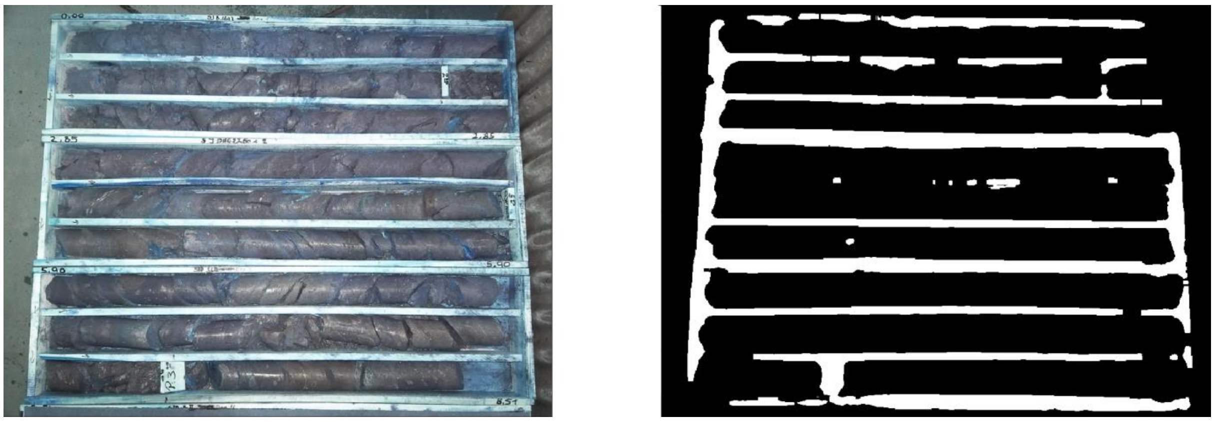
Figure 4 *Initial result from deep learning algorithm*

Conclusions

Based on the results, it is possible to identify the core trays using deep learning approach. U-Net had better performance than CNN. If we have more data for training we will get even better results at the end. It is also recommended to expand pre-processing steps and apply other filters like band-pass and median filters to remove more noise from original images.

Acknowledgement

We would like to acknowledge all of GoldSpot team.

References

U-Net: Convolutional Networks for Biomedical Image Segmentation by Olaf Ronneberger, Philipp Fischer, and Thomas Brox, arXiv:1505.04597v1 [cs.CV] 18 May 2015.

An introduction to Convolutional Neural Networks by Keiron O'shea and Ryan Nash, arXiv:1511.08458v2 [cs.NE] 2 Dec 2015.

Convolutional Neural Networks for Cellular Segmentation, Master's thesis by Sten-Oliver Salumaa University of Tartu.

Keras: The Python Deep Learning library, [Online]. Available: https://keras.io/. [Accessed 14 05 2018].

Splash of Color: Instance Segmentation with Mask R-CNN and TensorFlow, [Online]. Available: https://engineering.matterport.com/splash-ofcolor-instance-segmentation-with-mask-r-cnn-and-tensorflow-7c761e238b46. [Accessed 21 05 2018].

Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow, Matterport, [Online]. Available: https://github.com/matterport/Mask_RCNN. [Accessed 14 05 2018].