Detection of Mineral Deposits in NASA *LandSat* Images Using Convolutional Neural Networks

Sean McAuliffe, V00913346 **Spencer Davis, V00759537** **Kiana Pazdernik, V00896924**

Mateo Moody, V00918050 **Chris Wong, V00780634**

Abstract

We trained Convolutional Neural Networks (CNNs) to identify locations likely to contain useful mineral deposits on the Earth's surface using NASA Landsat Earth observation images. Training labels for supervised learning were generated programmatically using a corresponding dataset of known mineral deposits. AI assistance in locating mineral-rich locations can potentially help to minimize the environmental impact caused by prospecting and mineral extraction, which is generally harmful to the environment and human health. The Landsat dataset is preprocessed by downsampling to a resolution of 512x512. The images are labelled using a K-D Tree Search Algorithm into the minerals dataset. Both binary classification and regression labels were generated for all images. Sets of labelled images are organized into buckets based on their location. Ocean masking is performed to remove buckets over open water to prevent the resulting ML agent from simply learning to distinguish between land and sea. The images at each location are then sorted by degree of cloud cover, allowing for the selection of training images with minimal cloud cover. Once the images are preprocessed, they are split into training and testing sets for supervised learning. Several combinations of model architectures and hyperparameters were investigated to find a model which achieved high accuracy. For evaluation purposes, all models were compared against a baseline obtained by training the model on a training set with randomized labels. The trained models obtained an accuracy of 73% on the binary classification problem, and a MAE of 0.117 on the regression problem; these values both represent notable improvements over baseline. Training on more powerful hardware and using the full-resolution images may produce further improvements.

1 Introduction

1.1 Problem Description

The work of prospecting for mineral deposits is complicated and can be dangerous. In some cases it can be harmful to the environment and human health. Current techniques depend on specific geological knowledge, and utilize a combination of magnetic, gravimetric radiometirc, and seismic methods. Our work proceeds from curiosity; can AI assistance be used to learn generalizable patterns — as of yet unknown to human geologists — that emerge in surface features, which may be highly predictive in identifying useful mineral deposits?

1.2 Approach

This project was conceived, in part, to take advantage of the extensive Landsat dataset, which is freely available and contains over 8 million Earth observation scenes taken over the past 50 years [1].

Project GitHub: https://github.com/SeanMcAuliffe/satimg-classifier

Landsat images are very high resolution, as such, Convolutional Neural Networks were chosen as the primary learning agent for this project.

Convolutional Neural Networks have proven to be very effective in image classification tasks [2]. Of particular interest to this project is the ability of CNNs to quickly reduce the resolution of an image, while preserving learned patterns. This is useful for our project, as the Landsat images are very high resolution, and would be difficult to train on using a consumer GPU.

The Landsat dataset was combined with a dataset of known mineral deposits to generate training labels for supervised learning; the label of each image indicated the presence or richness of minerals in the image.

Much of the work involved in this project was in the preprocessing and problem setup. These steps anticipated and attempted to remove sources of error, and to constrict the the problem such that the ML agent would learn to identify useful patterns. For example, algorithms were devised to identify and select training images having a minimal level of cloud cover, and to exclude images taken over water. The full details of the preprocessing pipeline can be found in section 2.

1.3 Goals

We aim to train Convolutional Neural Networks to achieve 80% validation accuracy on the binary classification problem, and a Mean Absolute Error of ≤ 0.1 on the regression problem.

2 Problem Forumulation

The problem formulation step encompases the bulk of the work. This step prepares the data for the model to train on. The particular ways in which the dataset is chosen, labelled, and prepared effect what the model will learn. These decisions are made with the goal of constraining the problem in such a way that the model will learn to solve the problem as formualted in our minds. Each formulation rests on a particular philosophy of what should be learned, and each has its own strengths and weaknesses.

We formulated the problem in two ways: binary classification (2.2), and regression (2.3). Each is described in the following subsections. Both approaches shared much of the same acquisition and preprosccesing pipeline as described below.

2.1 Data Acquisition & Preprocessing

We acquired the entire set of approximately 45000 band 7 images from Landsat 4 of the public Landsat dataset on Google Cloud Platform. We chose band 7 (2.09-2.35μm) because it uses a Thematic Mapper sensor to capture light in frequencies optimal for identifying hydrothermally-altered rocks associated with mineral deposits. These images are each approximately 7500x7500 pixels in a single colour channel, and are taken at regular locations on a 2x2° latitude-longitude grid.

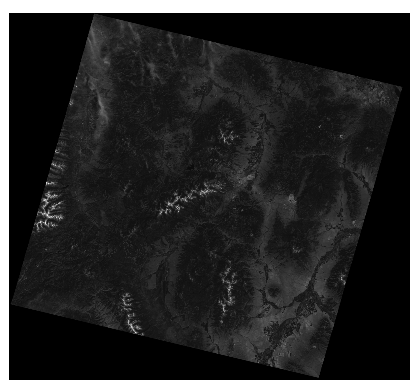
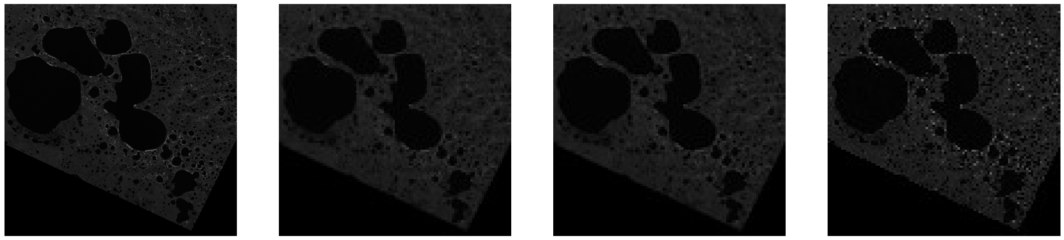


Figure 1: *An example Landsat image in band 7.*

We also acquired a dataset of approximately 350000 mineral deposits, each including the deposit's coordinates and the types of minerals present.

We automated the image downloads using a bash script. This script used the bash utility *ImageMagick* to downsample each image from 7500x7500 to approximately 512x512, to allow for training on consumer hardware. The corresponding metadata file for each downloaded image was also obtained, containing information such as the coordinates of the image corners. The download took approximately 6 seconds per image, for a total download time of approximately 75 hours.



(a) *Original* (b) *Rescaled* (c) *Resized* (d) *Downsampled*

Figure 2: Three compression techniques applied to the same Landsat image

Several differen algorithms from were experimented with: resizing, rescaling, and sampling. Resizing and rescaling use similar iterative processes of averaging neighboring pixels to effectively compress the image while reducing quality. Sampling is a more direct approach which simply selects a subset of the pixels evenly from across the input image. This is the lossiest of the three methods, but is also the fastest. The results of these three methods are shown in Figure 2.

Labels were created for each image using an algorithm accelerated by a k-d tree of mineral deposit locations; the k-d tree was used since k-d trees provide significant speedup for queries on spatial data [3]. Efficient label creation was nontrivial since it required identifying which mineral deposits were present in each image, and the images were neither axis-aligned nor perfectly rectangular. To address this, the label-creation algorithm first computed the enclosing circle of each image, then queried the k-d tree for the set of deposits present in the circle; brute force was then used to test which of these deposits were within the convex polygon of the image. This method provided a speedup of 20x over the naive approach.

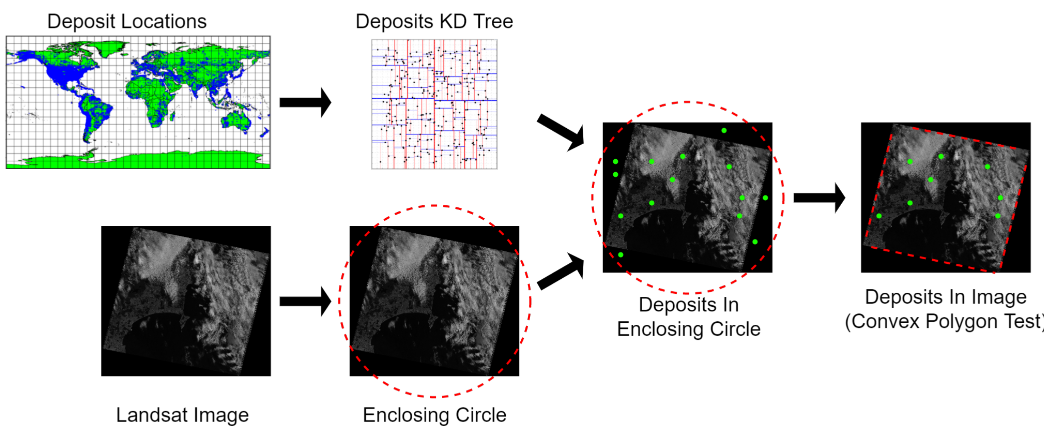


Figure 3: Label creation pipeline

Having obtained the set of deposits present in an image, separate labels were created for binary classification and regression. The binary classification labels indicated whether any minerals were present in the image, and the regression labels indicated the mineral richness of the image. The richness score was computed as the log transform of the absolute count of deposits in the image, normalized to the range 0-1. The log transfrom was used to reduce the skew of the regression values.

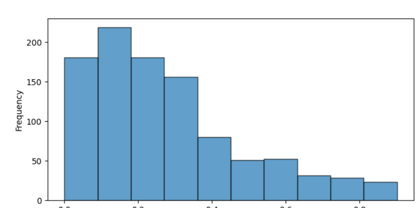


Figure 4: Distribution of log-normalised richness scores

$$label'_i = \frac{\ln(label_i + 1)}{max(labels)} \tag{1}$$

Location Buckets: Since Landsat images are taken at regular intervals, and our dataset contained several images at each location, it was necessary to ensure that no location appeared in both the training and testing sets simultaneously. To achieve this, the images were grouped by location, resulting in approximately 6000 unique location buckets.

Cloud Cover: Having grouped the images by location, the images in each bucket were sorted by average pixel brightness, which for band 7 images effectively sorts by degree of cloud cover. This permits the selection of training images having less cloud cover.

Ocean Masking: Finally, images in location buckets over the ocean were removed, to ensure that our models did not learn to simply detect land. The Python library Geopandas was used to load a geospatial dataset containing the polygons of the boundaries of all land on Earth; this dataset was then queried to determine whether each image's centerpoint was over land.

Training + Testing Sets: To prevent our models from simply learning the underlying distribution of training labels, we ensured that the ratio of class labels in the training set is balanced. In binary classficiation this is trivial, we enforce that the training set be comprised of 50% positive and 50% negative labels. In the regression case, the set of all examples has some underlying distribution shown in figure 4. The training set is then sampled randomly (without replacement) from this distribution, and therefore comes to have a very similar distribution. The remaining images are used as the validation set.

The training + testing set was drawn from the set of all images such that at least one image from each location bucket was included. The training set was then randomly sampled from this set, and the remaining images were used as the testing set. All experiments used a 80/20 split between training and testing sets.

Padding: Finally, since the Landsat images vary slightly in size and are not perfectly square, each image was padded to 512x512 pixels with black pixels, and all pixel brightnesses were normalized to the range 0-1.

2.2 Binary Classification

The binary classification problem setup was the first to be designed. It represents the simplest formulation of the problem, and our naive initial understanding. The primary goal of this problem was to determine whether a model trained on the dataset described above could learn anything relevant to the problem which would enable it to perform better than chance. To this end, examples with binary labels were prepared for the model as described above.

This approach initially proceeded from the assumption that mineral deposits would be sparesely located throughout the world, and that their presence might correspond to certain easily detectable surface features. However, when the labelled location buckets were later plotted on a world map it became obvious that there was a clear preponderance of positive labels over land. Given the large surface area covered by each image almost all images contain at least some small number of desposits.

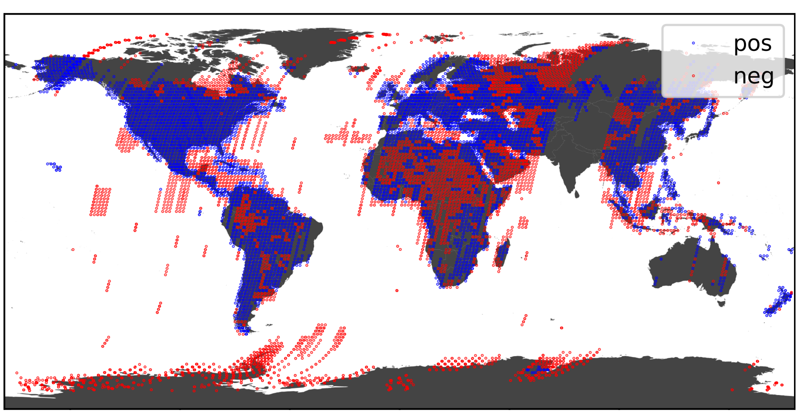


Figure 5: All labelled location buckets

Additionally, negatively labelled images do not necessarily mean that there are no mineral deposits in the region. In fact, this is a major philisophical problem with the problem setup. Many of the false positives that the model will produce may in fact be true positives representing new desposits which the model has successfully identified. Without any way of independently verifying the presence of these deposits we are unable to truly evaluate the model's performance, and the model could never further learn from these examples.

Nevertheless, this problem formulation does allow us to successfully determine whether our dataset contains information which is learnable by a CNN.

2.3 Regression

The regression problem was designed to address the issues with the binary classification problem, and sought to investigate the feasibility of the practical application of our methods. In practice, a predicted richness score is more useful than a binary prediction when choosing a location to dig for minerals. The regression formulation also addresses the problem of whether negative examples are really negative: if a regression model achieves a low error when trained only on examples containing known deposits, then predicts a high richness score on an example containing no known deposits, it may indicate the presence of undiscovered deposits.

To improve the interpretability of our regression model performance, the training and testing sets for the regression problem were sampled such that each had a near-equivalent distribution of label values.

3 Background and Related Work

Much of the previous work done to integrate satellite and aerial photography into the mining industry has been to better assess geographic factors relevant as obstacles to construction and extraction [4]. As well as to predict the extent of damage down to the natural landscape, for instance, to determine if much tree cover would have to be removed. Other work at automatically detecting minerals from orbit has been limited by the sensor capabilities of the satellites, indeed typically only those sensors with very high resolution and particular imaging wavelengths are useful for geological applications [5].

Other work has incorporated AI in the development of maps, and to search for particular features and reflectances which are already known to geolosits to correlate with particular kinds of despits [6].

Our work proceeds from the question; can new patterns useful in predicting the location of deposits, as of yet unknown to geologists, be learned by ML agents? Our work is limited by the resolution of the data and our computing power. A more exhaustive search of this question would train on multiple EM bands, and the full resolution available; and would likely achieve more impressive results.

4 Methods

This section describes the experimental methods used to train the CNNs, and tune the CNN architecture and hyperparameters. To reduce the size of the search space of possible architectures and hyperparameters, a heuristic was applied: first a set of candidate architectures were trained and evaluated; then only the architecture exhibiting the best performance was subjected to hyperparameter tuning.

4.1 Model Architecture

Convolutional Neural Networks are a type of deep neural network which are well suited to image classifications tasks. They are composed of a series of convolutional layers designed to quickly extract ever-higher level patterns from the data. The convolutional layers are followed by a series of dense layers which perform the classification task. The final layer is a single ouput neuron which produces a value in the range 0-1. In the binary classification case this is taken to be the predicted probability that an image contains at least one mineral deposit. In the regression case this is taken as a prediction of the extent of mineral richness.

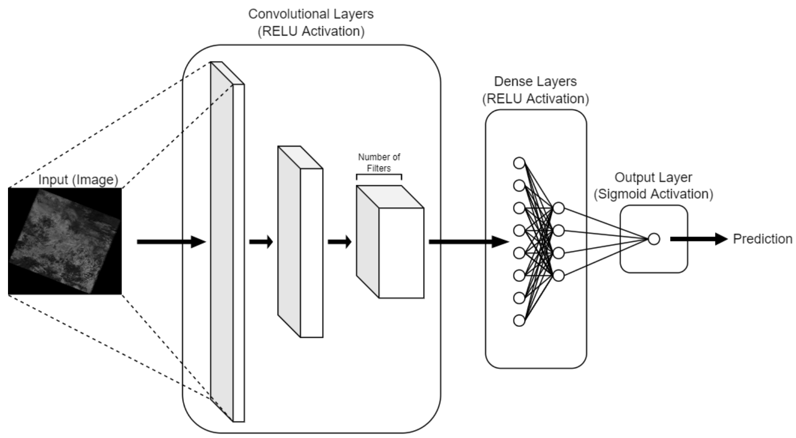


Figure 6: Generic CNN architecture

Convolutional layers each have assosciated filter(s). Each filter is an fxf grid of weights, and an assoscitad bias. A convolution operation involves "sliding" the filter over the layers input (for the first convolution layer the input is the example image itself). The amount by which the filter is slide at each step is determined by the stride. For each location of the filter against the input layer, an element-wise multiplication is performed and summed to produce a single output fed into the following layer. Equation 2 describes the 2D convolutional operation in the case that the stride is 1 in each dimension. Where Z is the convolutional output, W is the matrix of filter weights, X is the input layer, and b is the bias.

$$Z_{i,j} = \sum_{m=1}^{f} \sum_{n=1}^{f} W_{m,n} \cdot X_{i+m-1,j+n-1} + b \tag{2}$$

Several nonlinear activation functions are available to be applied to the output of each convolutional layer. The most common, and most successful in this project was the Rectified Linear Unit (ReLU) function. The ReLU function is defined below in equation 3.

$$ReLU(x) = \max(0,x) \tag{3}$$

It is common to also include a pooling layer after each convolutional layer. Pooling is a technique which can reduce the resolution of the output from a previous layer, without introducing any further trainable parameters. This simplifies further training while preserving the most important features of the input. Commonly, max pooling layers are used, their behaviour is described in equation 4.

$$P_{i,j} = \max_{m=1}^{f} \max_{n=1}^{f} X_{i+m-1,j+n-1} \tag{4}$$

To prevent overfitting, L1 and L2 regularization can be applied to the weights of each layer. These regularizations are penalties added to the models loss function to discourage large weights. This prevents the model from depending on any particular features too much. L1 regularization is less agressive, it is described in Equation 5. L2 regularization is more aggressive and is described in Equation 6. In both cases λ is a hyperparameter which controls the strength of the regularization.

$$L1(W) = \lambda \sum_{i=1}^{n} |w_i| \tag{5}$$

$$L1(W) = \frac{\lambda}{2} \sum_{i=1}^{n} w_i^2 \tag{6}$$

We additionally found that batch normalization was useful in increasing model performance and possibly improve training times. Batch normalization is a technique which normalizes the output of each layer to have a mean of 0 and a standard deviation of 1 according to Equation 7. This has the effect of preventing any particular layer from dominating the training process.

$$\hat{X} = \frac{X - \mu}{\sqrt{\sigma^2 + \varepsilon}} \tag{7}$$

The candidate architectures each had a unique combination of two architectural characteristics: the complexity of the convolutional network, and the complexity of the dense network. "Big", "medium", and "small" configurations were defined for the convolutional and dense networks, and every possible combination was trained to convergence, for a total of 9 candidate architectures. Since the convolutional network extracts abstract features of the image, and the dense network performs classification using those features, varying these architectural characteristics independently allowed us to investigate the nature of the patterns relevant to the problem. For example, if a deep convolutional network provided no improvement over a shallow one, it could indicate that the general, high-level features of Landsat images carry little information relevant to the problem of mineral richness.

The following table specifies the configurations tested:

4.2 Hyperparameter Tuning

4.3 Model Evaluation

Prior to the architecture and hyperparameter tuning, performance baselines were obtained for the binary classification and regression models; this was an important preliminary for evaluating later model performance, since it allowed us to measure the improvements provided by each optimization. These baselines were obtained by training the medium-medium CNN configuration on examples having randomized labels for the training set, to ensure that the training set contained no learnable information relevant to the problem. The binary classification baseline was 50% validation accuracy, and the regression baseline was a MAE of 0.127.

The preliminary baseline, without any optimizations or hyperparameter tuning, and with the training labels randomized, returned results for binary classification of 50 value accuracy, and the regression produced a mean absolute error (MAE) of 0.127, which is a measure of the difference between the predicted and actual values. A lower MAE indicates better accuracy in predicting the richness score of the mineral deposits. Obtaining a baseline was an important step in evaluating the performance of the models, as it helps determine if any additional efforts to optimize the model are improving its performance.

5 Results

5.1 Binary Classification

5.2 Regression

6 Discussion

6.1 Limitations

These results are limited by the ambiguous quality of the mineral deposit dataset, which is incomplete. And although we manually verified the locations of a random sampling of the deposits, we did not exhaustively verify the dataset. Any errors in the minerals dataset would be expressed as inaccuracte labels in our dataset, and would reduce the reliability of our models. And since some of the mineral deposits are located at open mines visible from space, our model may be learning to simply detect mines, to some degree. Our results are also limited by the consumer hardware on which we trained our models; although we saw little performance benefit from larger models, much larger models and more powerful hardware may still obtain better performance even on our downsampled dataset.

6.2 Future Work

7 Conclusion

References

[1] Landsat

[2] https://towardsdatascience.com/using-convolutional-neural-network-for-image-classification-5997bfd0ede4

[3] https://opendsa-server.cs.vt.edu/ODSA/Books/CS3/html/KDtree.html

[4] https://www.satimagingcorp.com/applications/energy/mining/

[5] https://www.esri.com/about/newsroom/arcwatch/mineral-exploration-in-the-hyperspectral-zone/

[6] https://arxiv.org/pdf/2103.07678.pdf