# Classifying Mars landmark data using convolutional neural networks

Lewis Sword*
*Queen Mary University of London*
*Centre for Theoretical Physics*
(Dated: March 13, 2023)

This report details how convolutional neural networks (CNNs) may be utilised to classify Landmark image data of Mars' surface. We train CNNs for two different classifying tasks and evaluate their success given a choice of architecture and parameters.

## I. INTRODUCTION

Convolutional neural networks (CNNs) are an effective way of classifying images. Based on the typical multilayer perceptron construction, CNNs introduce additional "convolution" layers which apply filters to the input data. Used in conjunction with pooling, dropout and hidden layers, CNN models allow predictions to be made on unseen images and are hence the subject of this report.

The input for the study is the HiRISE Mars landmark image data courtesy of [1] and the network itself will be generated using the TensorFlow platform in Python. This original data consists of 70301 images with pixel dimensions $(256\times256)$. These landmark images are divided into eight categories: "other" , "crater", "dark dune", "slope streak", "bright dune", "impact ejecta", "swiss cheese" and "spider", which are classified by integer labels $\{0,\ldots,7\}$. The aim of the report is divided into two tasks. Task 1 (full set image classification) is to train and test a CNN model on classifying all eight landmark class categories individually, while task 2 (binary classification) aims to characterise two subsets from these class categories — dunes and all other classes — using binary classification.

The outline is as follows. Section II presents the CNN architecture of both tasks and describes their implementation in detail, explicitly defining the parameters used. Section III begins by providing the analysis for the full set image classification task CNN. Here we first present a figure representing the differences in train and test accuracy of the model as a function of epochs. A general summary of how well images from all classes are predicted by the model is then captured by a confusion matrix. This is followed by a plot of twenty example images from the test data set, allowing direct comparison between their true and predicted classification labels. Next we inspect the success of the binary classification model, again starting with an accuracy vs. number of epochs plot and ending with a receiving operating characteristic (ROC) plot. Finally, in Section IV we provide an overview of the project, summarising results and provide possible ideas for future model development.

* l.sword@qmul.ac.uk

## II. IMPLEMENTATION

Given that tasks 1 and 2 have different objectives, we now describe the implementation of each CNN structure and their alternative sets of parameters.

The set-up for task 1 starts with balancing the data. Category 0 - "other" contains significantly more data than the remaining categories and hence we choose to limit this class' data to 5000 which is on par with next largest data category. In total the model for this task utilises 16977 images. After this preprocessing we feed the image data from all 8 categories into the CNN, using a 70:30 train to test split. The convolutional neural network employed for this task features a concise sequence of various layers, the architecture of which can be seen in figure 1.

The first layer is a lambda layer which serves to resize the image data. This was utilised due to the computational constraints, and we chose to resize the pixel dimensions of each image to $(60 \times 60)$ [2]. At this resolution, salient features are still apparent by eye. The following layer is a convolution layer with 32, $3 \times 3$ sized filters which is followed by a $2 \times 2$ pooling layer. The filters of convolution layers aid in identifying patterns/features in the images by producing feature maps. To help reduce complexity in the network, a dropout layer is then installed, serving to "turn off" a fraction of randomly selected elements in the feature map (see Table I for the value). Another convolution-pooling-dropout layer sequence is then added, this time changing the convolution layer to have 64, $5 \times 5$ filters. The data is then flattened and processed by a 256-neuron dense layer. Each convolution and dense layer used, is equipped with a ReLu activation function: useful since it removes vanishing gradient issues. Finally we reach the output layer, which contains 8 neurons (to match the 8 classification categories) and a softmax activation function. This final activation function serves to provide prediction results as an array of probabilities. Note that the loss function used in evaluation was categorical cross-entropy. The parameters selected during training of the model are given in Table I.

For task 2, the binary classification task, the images were reassigned class labels according to the required subset selection: "all dune images: 1" and "all other classes: 0". This data set was then balanced such that each of the two classes had approximately the same number of

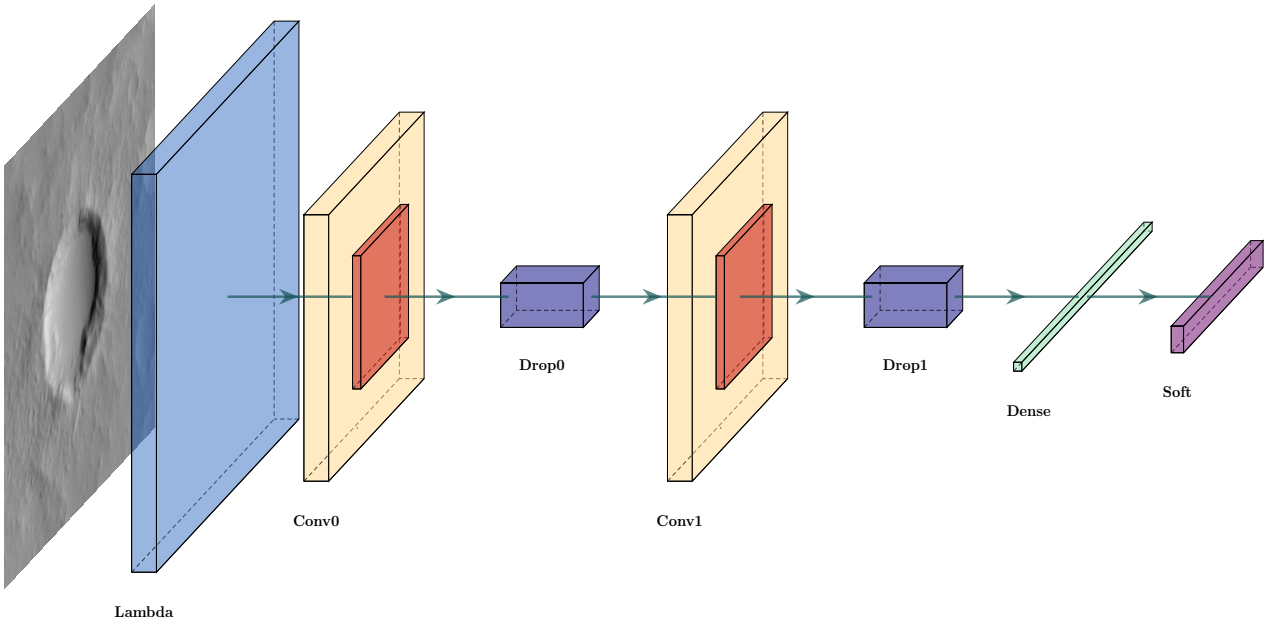


FIG. 1. The figure demonstrates the simple CNN architecture used for the full image set classification of task 1. The CNN includes: a Lambda layer that applies pixel resizing, coloured light blue; convolution layers Conv0 and Conv1 that apply filters, coloured yellow; pooling layers, coloured orange; dropout layers Drop0 and Drop1 that apply regularisation, coloured dark blue and a Dense layer, coloured green. The final Soft layer, coloured purple, features a softmax activation function which provides a probability for each output class prediction.

images, totalling to 5891 images.

| *Task 1: Full set image classifier* | |
|---|---|
| **Parameter** | **Value** |
| Test size | 0.3 |
| Pixel dimensions | $60 \times 60$ |
| Learning rate (Adam optimiser) | 0.00085 |
| Dropout | 0.8 |
| Number of epochs | 20 |
| Batch size | 30 |

TABLE I. Parameter table for the full set image classification task.

The train and test data were again split as 70:30. As for the network structure, this was kept exactly the same as task 1, with the exception that the final dense Soft layer was replaced by a single neuron layer with sigmoid activation function. The sigmoid activation function allows us to categorise images between either class by setting a certain threshold value (a variety of these thresholds are used in Figure 6, for example). The loss function used for this task was binary cross-entropy. Finally, the parameter values for this task are listed in Table II, and the dropout layers in each separate task all take the same value as listed in their respective tables.

| *Task 2: Binary classifier* | |
|---|---|
| **Parameter** | **Value** |
| Test size | 0.3 |
| Pixel dimensions | $60 \times 60$ |
| Learning rate (Adam optimiser) | 0.0004 |
| Dropout | 0.86 |
| Number of epochs | 20 |
| Batch size | 30 |

TABLE II. Parameter table for the binary classification task.

## III. CNN EVALUATION AND PREDICTIONS

### A. Task 1: Full set image classification

Having trained the model using the CNN architecture of Figure 1 and the listed parameters of Table I, we can now evaluate its success in classifying the eight individual types of Mars landmark images. Firstly, we study the model accuracy plot in Figure 2. Here we see that the train and test/validate accuracy generally both grow together with increasing epoch number. The validation accuracy curve has a somewhat oscillatory form. This could potentially be dampened by further reducing the learning rate, increasing batch size and then running over a greater number of epochs to compensate these changes. At later epochs the train accuracy begins to slightly outperform the validation accuracy. This indicates that the

model may be overfitting to the training data, and means that potentially by introducing more regularisation such as increasing the dropout parameter value, we could see better results. With this parameter selection, the accuracy on both sets is above 70% during the last 5 epochs which, given the reduction in data compared to the original dataset (due to the imposed balancing), seems sensible.

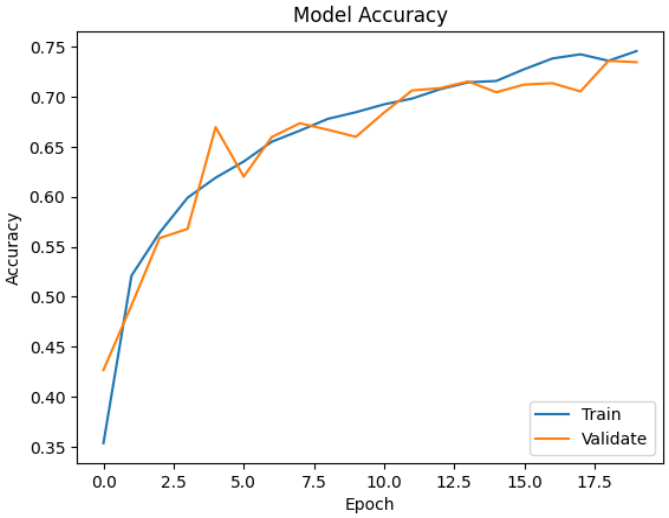


FIG. 2. Training and test (validation) accuracy curves as functions of epoch number. Generally the accuracy of both train and test/validation data sets grow at a similar rate with epoch number, reaching $\sim 70\%$ in the last epochs.

Another natural way to analyse the data is to compare all the different outcomes of the model predictions to their true values. An efficient and visually simple way to do this is by using a confusion matrix, as seen in Figure 3. To help understand the information portrayed by the plot, we study a short example looking at a particular true label/row. Take the row for true label 4 (bright dune) for instance. The box with the largest value 0.88 (and box which is darkest blue) in this row corresponds to predicted label 4. This implies that upon processing all the data, the most frequent prediction for class 4 images was indeed, class 4. This information, in conjunction with the small values in the remaining row boxes, means there is not much "confusion" in the models' predictions for bright dunes. Applying this idea to all boxes in the confusion matrix, we can see that generally, the model does quite well since the diagonal entries (true positives) all score quite highly. Additionally, it makes sense that given the variety of features in the class 0 ("other") images, it is somewhat likely that the model will predict that an image belongs to this class, when in actuality it belongs to another (this can be seen from the non-negligible values in column zero).

The final analysis of the full set image classification task comes from observing the actual images associated with the model predictions. Figure 4 provides a

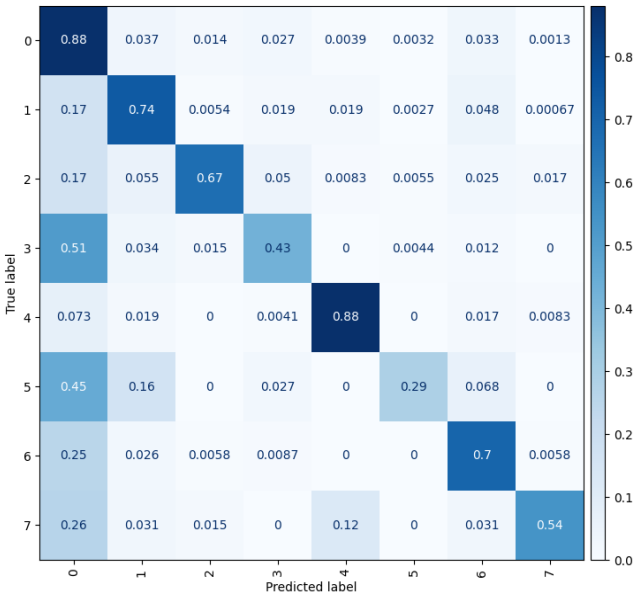


FIG. 3. A normalised confusion matrix generated from predictions made on test data of the CNN. The diagonal elements tend to produce large "heat" values, indicating that the CNN model yields predictions close to the true values.

small subset of the landmark images, specifically plotting twenty with both their true (T) and predicted (P) class labels. The samples which predict the correct true label appear to have distinctive features, such as the "bright dunes" or "swiss cheese". Clearly these unique patterns, which are readily identifiable by human observation, are also successfully picked up by the model. Images like "slope streak" however, seem to be slightly less easy to predict, owing to similar/shared characteristics with other classes.

### B. Task 2: Binary classification between image subsets

Task 2 investigates binary classification by grouping the bright and dark dune images into a single class, and the remaining original class images into another. Adopting the parameter values seen in Table II and refining the CNN to accept two classes whose predictions are determined by a single choice (0 or 1), the success of the model is assessed using an accuracy vs. number of epochs plot as well as a receiver operating characteristic (ROC) plot.

The accuracy vs. number of epochs plot in Figure 5 shows how well the model performs on the training and validation data sets. One clear feature of the plot is the disparity between the curves at approximately epoch 5. From this point, the training accuracy far exceeds the validation accuracy until epoch 10, hence this region experiences overfitting. However, afterwards they both slowly grow at the same rate, with slight oscillations in the validation curve. Both curves achieve accuracy val-

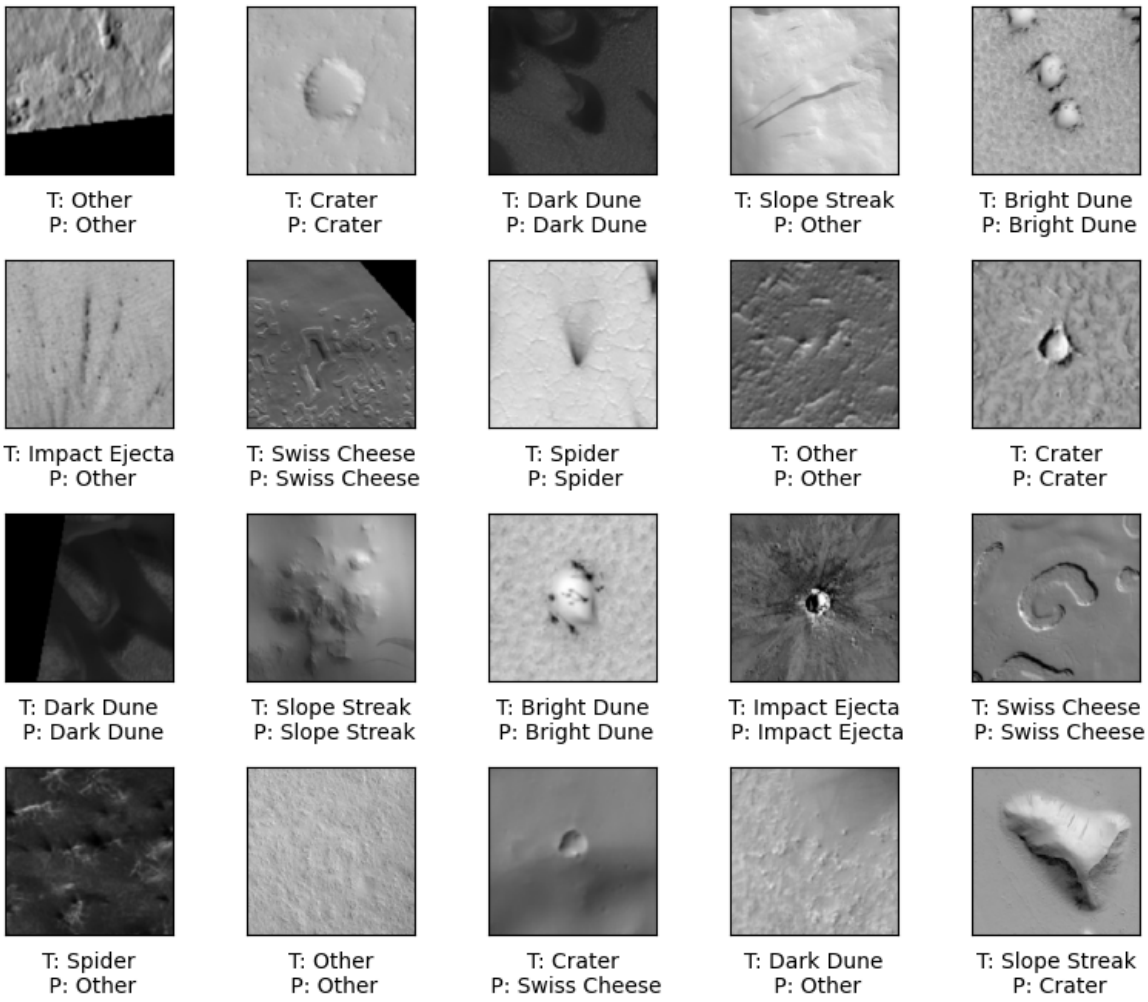


FIG. 4. Twenty unique, (256 × 256)-pixel example images consisting of at least one from each classification category. The image plots act as visual verification as to where the model has picked up on various salient features, and where it has not by comparing their true labels (T) and predicted labels (P).

ues of around 86% in the final few epochs, so the model is relatively successful. Some factors to be considered in the model performance are the amount of input data and the complexity. The total number of images processed after balancing, 5891, is rather small compared to the initial HiRISE set, and we may expect to see better results by increasing this number. The binary classifier model was also heavily dependent on learning rate and dropout parameter choices, and thus further fine tuning of those could also improve the smoothness of the validation curve when combined with a larger number of epochs.

The plot in Figure 6 demonstrates the ROC curve. When we predict with the model, the output due to the sigmoid function will be a number between 0 and 1. To classify which image belongs to the bright and dark dunes set or to the set containing all other image categories, a threshold value is applied to this prediction output. The ROC curve plots the performance of the model at all possible threshold choices and indicates just how distinguishable the two classes are. A useful feature of the ROC curve is the Area Under the ROC Curve (AUC) value, which is a scalar measure of model performance across all possible classification thresholds. A good ROC curve is one that traverses near to the top left corner such that the rate of finding true positives will become large at a small false positive rate. This also means a curve that produces an AUC value close to 1 (since the axes are normalised to 1, a perfect/ideal ROC curve will have AUC=1). For our model, the ROC curve indeed tends close to the top-left and has an AUC value of 0.94. This presents further evidence that the implemented binary classification model works well.

## IV. CONCLUSION

In both the full image set classifier and binary classifier tasks, the CNNs employed were able to make successful

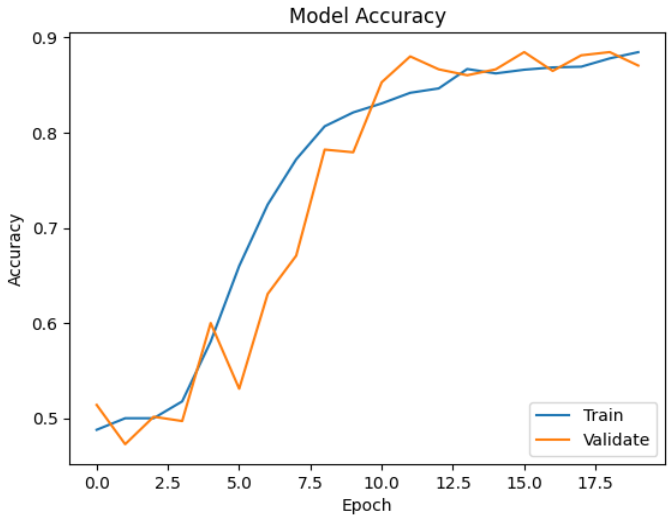


FIG. 5. Model accuracy as a function of number of epochs for the train and validation sets in the binary classification task. During the first 10 epochs, there is some overfitting, made clear by the better train accuracy. Generally the curves are similar past epoch 12, obtaining accuracies upwards of 80%.

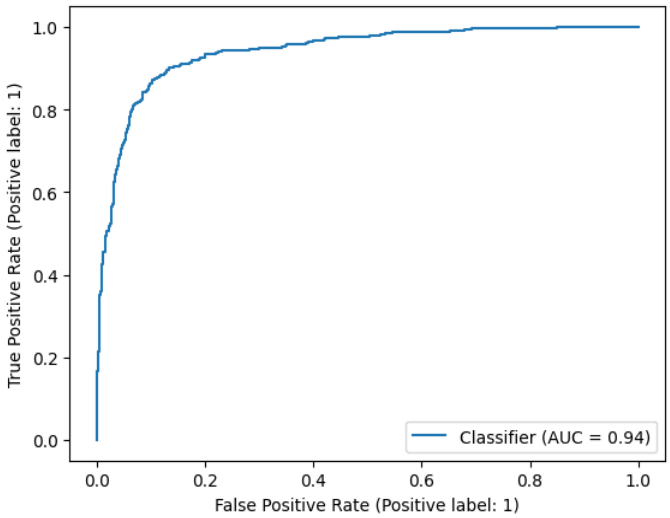


FIG. 6. A plot of the receiver operating characteristic (ROC) curve for the binary classification task. The AUC=0.94 being close to 1 indicates that the model is successful at distinguishing between the two classes.

predictions on unseen Mars landmark images. Task 1 achieved a final epoch accuracy of 0.7346% while task 2 had an accuracy of 0.8705%. Given these values, additional modifications could be made to the setup that may lead to an increase in validation set accuracy and reduction of loss. Quoting [3], "the learning rate is perhaps the most important hyperparameter", and hence a more in depth study of its effects would likely lead to better predictions. We could also dynamically update the learning rate using a TensorFlow "learning rate scheduler" to make the model more efficient. A general change in architecture could also prove fruitful: including different convolution layers, pooling and filter sizes, as well as alternative numbers of neurons in dense layers. At the cost of additional computation time, increasing the pixel size at the lambda layer will increase the data available and potentially lead to better learned results. In turn this also provides more freedom in the filter size selection of the convolution layers. For example, applying a third convolution layer, with a slightly larger filter size (say $7 \times 7$) and larger number of filters could lead to a greater learning of the global structure in the images. Another parameter to explore is the number of epochs. While the tasks saw some instances of overfitting, after fine tuning other parameters to help regularise, extending the number of epochs could plausibly produce greater accuracy for our classifiers. Finally, batch size was chosen to be relatively small to assist in efficiency, but like the other parameters, modification is also an option.

[1] You Lu and Kiri Wagstaff. Mars orbital image (hirise) labeled data set, November 2017. URL `https://doi.org/10.5281/zenodo.1048301`.

[2] Note1. Note that the image data features an RGB dimension of length three. The components all take the same value however, hence these images can be considered grayscale.

[3] Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, Cambridge, MA, USA, 2016. `http://www.deeplearningbook.org`.