Building a CNN for Moonquake Classification

Savannah Cofer
cofer@stanford.edu

Weston Kirk
wkirk@stanford.edu

Gary Xu
garyxu@stanford.edu

Abstract

Machine learning approaches are increasingly being applied to lunar seismic data for accurate and fast classification, especially in the context of efficient processing and storage of seismic data on future lander missions. In this paper, we create a 2D CNN classifier to classify moonquakes into one of four major types, and applied it to identify currently unclassified moonquakes which warrant re-evaluation by experts for potential labeling. Two to five layer models are tested on separate subsets of LSPE data over 200 grade A moonquakes classified as deep, impact, and shallow, and nearly 3,000 grade A, B, and C classified moonquakes, achieving an average accuracy of over 90% *on the grade A subset and* 80% *over the entire dataset. We also discuss and intially quantify the possibilities of a distinct and separate subclass of moonquakes.*

1. Introduction

1.1. Problem

We want to be able to comprehensively classify Apollo moonquake data to refine our understanding of the internal structure of the moon, and determine if modern machine learning techniques can help us in develop statistical models to represent classification, contribute to automatic classification on future lander missions, and filtering traces for data prioritization [1]. Specifically we seek to determine the utility of varied combinations of input data and the effectiveness of neural net architectures to classify seismological data. The original data is difficult to classify due to high levels of noise [2], a common problem also seen in seismic data from Mars whch there have been attempts to resolve [3].

1.2. Prior Classification Efforts

Prior studies, such as Nakamura et al. 1982 [4] have done initial classification of the detected moonquakes throughout the Apollo missions. However, the majority of these early classifications were done manually which is both time consuming and not feasible at a large scale; and, more than half of the original moonquakes were unclassified at this time. Since then, computational analyses have aided in classification using techniques such as cross correlation [2] [5], and more recently spectrogram data has been used to train supervised learning methods and convolutional neural networks (CNNs) to detect seismic events [1].

1.3. Alternative Parameters to Spectrograms

Other methods have emerged including using thermal data or lunar orbital parameters with time to locate deep moonquake source locations [6], but there has been no comparison or combination of such methods yet. The accuracy of supervised learning classification for seismic data still leaves much to be desired, reaching only around 70% in multiple studies [6] [7], with certain models performing much better [1] [7].

1.4. Proposed CNN Model

We introduce a classifier trained on a three grades of data (A, B, and C), in an attempt to recognize wave characteristics [1] [5] using a CNN, without overgeneralizing, with the potential to add lunar orbital parameters and time data [8], and perhaps thermal data to later models. With this, we intend to build a model which classifies moonquakes into one of three main types, being deep, shallow, or impact moonquakes, discussed in more detail in section 2.3. We additionally use this methodology to identify unclassified moonquakes for potential classification by experts. Currently, the dimensionality of spectrogram inputs are an area of conflicting research, with some papers finding success using 1D spectrogram input data [7], while others 3D spectrogram input data [1], or other methods using entirely different inputs [8]. Our model inputs will include seismometer data collected from the four station seismic network on the moon known as the Apollo Lunar Seismic Experiments Package (ALSEP) [4]. The model will be trained on previously classified moonquakes, including those catalogued in Nakamura et al., 1982 [4], which will comprise the validation set as well. Finally, we will test the model on previously unclassified moonquakes to determine its accuracy and make any additional observations as necessary.

1.5. Evaluate the Solution

To evaluate our method and model, we compare our classification accuracy of both long period and short period lunar seismometer data to that of previous studies, keeping in mind the 70% threshold mentioned earlier. We also evaluate the effectiveness of using various types of data including waveform data, lunar orbital parameters, and thermal data both in isolation and in combination. (Additional information and conclusions will be provided once we are able to produce real results). Ideally, we are able to classify currently unclassified moonquakes into one of the four major types [4]. In addition, there exists the possibility of the identification of subtypes within a known group of earthquakes [9]. We want to create a model which is able to classify earthquakes by picking up on details in the data that have been overlooked by previous studies, while remaining generalizable to future data, as is the goal of many neural net architectures.

2. Related Work

2.1. Methods and Characterization of Moonquakes

The characterization of moonquakes into several subcategories by seismicity and "types of natural seismic sources" [4] has been thoroughly explored and well defined. Those are most prominently, deep moonquakes, shallow moonquakes, thermal moonquakes, and meteoroid impacts [4]. However, there is not a universal, unchanging agreement on the classification of known moonquakes, and additional categories or subcategories may exist. In fact, the classification of moonquakes is constantly changing due to many reasons, including the evolution of methods (see Machine Learning Approaches below) and access to more complete and comprehensive data sets [2].

2.2. Machine Learning Approaches

With the relatively recent drastic improvements to Machine Learning with neural nets, many papers have utilized machine learning to now classify or help classify moonquakes in favor of earlier manual methods. It has even become possible to very accurately catalogue planetary seismicity even without local training data with the usage of convolutional neural nets [1] [10]. Techniques such as stacking enabled classification of previously unclassified deep moonquakes, yet struggle to classify novel events [5] [11]. However, neural networks can address limitations of template matching without sacrificing accuracy as evidenced by using a deep learning techniques and CNN to distinguish between P-waves, S-waves, and noise [11]. Many architectures have been tried for CNNs for moonquake classification with both 1D [7] and 3D input seismogram data [1], while other methods use entirely different inputs [8]. According to Stott et al. 2023, the optimal neural network has "the lowest validation loss and smallest gap between validation and training loss" [3] which is what we will aim to achieve.

2.3. Lunar Structure

One more category of interest that many papers cover on the topic of moonquakes is the internal structure of the moon - after all, one of the biggest reasons for studying and classifying moonquakes is the insight they give on the lunar structure [12] [13]. Our understanding of the moon's interior and systems are constantly changing. Initial analysis of moonquake data allowed for the recalculation of P- and S- wave velocity profiles, and more precise definitions of the upper and middle mantle. [4] [14]. Furthermore, a new seismic model of the moon has been proposed several times [15], new patterns in structure have been revealed through data analysis, such as the discovery of two previously undescribed belts of moonquake epicenter activity [4], and we are currently unsure of the main driver behind thermal moonquakes [9].

2.4. Other Applications

The study of moonquakes has many important and consequential applications - for example, the same methods of classification and characterization can potentially be applied to earthquakes. Currently, though not perfect, much progress is being made on the automated detection of earthquakes [16]. Re-framing existing data is also improving classification capabilities. For instance, thermal moonquakes have been classified into two different types [9], machine learning approaches have resulted in improved signal to noise ratios by considering seasonal atmospheric conditions [3], and using spectrograms as opposed to time-series data accounts account for noise differences in moonquake and earthquake data due to sensor sensitivity [1]. Moreover, with the machine learning methods mentioned earlier, it has even become possible to automatically pick up local earthquake seismic data [1] [10]. [9]

2.5. CNN Classification by Type

In summary, techniques to clean and consolidate data [9] [3], has enabled many machine learning methods, specifically CNNs to emerge for moonquake classification [7] [1] [8]. However, these models differ with input data and have not classified all of the originally 7633 unclassified moonquakes [4] using new data cleaning techniques, especially in under-analyzed areas of the moon where data exists. We can classify moonquakes using Convolutional Neural Network (CNN) we can classify moonquakes inton deep, shallow, and impact, with the end goal of identifying previously unclassified moonquakes from the Apollo missions. In summary, moonquake classification is still undecided for much

of the Apollo records, and CNNs are an emerging technique for classification.

3. Methods

3.1. Datasets

Our training and validation sets comprised of publicly available Apollo lunar seismometer data of classified moonquakes from the PSE and LSPE from Apollo stations 12, 14, 15 and 16. The training and validation data were labeled as deep moonquakes, shallow moonquakes, artificial impacts, and unidentified. We used previously despiked broadband data to train our model. We divided this dataset into training (80%) and validation (20%), and obtained previously unclassified suspected moonquakes cataloged by Nakamura et al. 1982 [4] for our test set. First, we only trained on Grade A data in order to assess the results on high-quality data. Then moved onto a robust training set comprised of Grade A, Grade B, and Grade C events, which is more representative of experimental data.

Our training and validation sets consisted of over 3000 previously classified moonquakes from the LSPE, including over 300 grade As, as depicted in Table 1. Note that for certain testing such as hyperparameter sweeping, we use a subset of this data for more efficient but equally accurate results.

Moonquake Label Dataset				
Grade	Deep	Impact	Shallow	Unlabeled
A	66	223	15	-
B	885	179	1	-
C	1658	213	0	-
U	-	-	-	1951

Table 1. Number of samples in our dataset given specific grades (A, B, and C), classifications (Deep, Impact, Shallow), and unlabeled data (U).

3.2. Input Spectrograms

Our multiclass classification model consisted of using a standard 2D-CNN using a colored spectrogram converted from Apollo lunar seismometer waveform data as inputs (**Figure 1**). As a general notion, spectrogram traces that appear more distinctive to the human eye will have more distinctive features, and in theory be easier to classify, as CNN architectures are "based upon visual process of humans" [17], providing a useful analog. In order to create spectrograms, we convert .mseed files and read in the data using functions from the ObsPy library. To visualize and plot the data, we used Matplotlib. We can adjust the resolution and no overlap parameters to see which combination produces visually crisp spectrograms. To do so, we changed two parameter outputs, with the first being the 'nfft', which refers to the number of data points in the fast Fourier transform (FFT). The default value is 256 and is used in powers of 2. The other parameter is the 'noverlap' value which represents the number of points shared between each block. The default value is 128 and needs to be a fraction of the nfft value (i.e. cannot have more overlapping pixels shared than pixels themselves). In order to test which combination of parameters worked best, we tested initially tested a range of nfft values centered around the default (64, 128, 256, 512, 1024) and for each tested three values for noverlap as multipliers of these values (0.25, 0.5, 0.75) for one trace of a Grade A deep moonquake. After this, we repeated this process using a smaller set of nfft values (64, 128, 256), with the same fractions on a Grade A shallow moonquake, Grade A impact moonquake, and Grade C Deep moonquake, to see how the waveform signatures responded across moonquake type and grade.

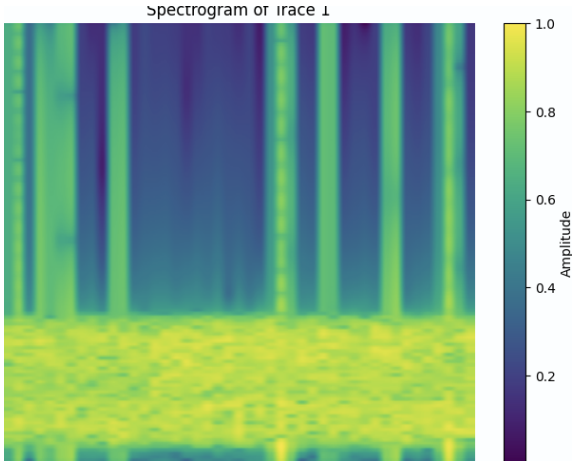

Figure 1. An example converted, colored spectrogram for a deep moonquake

3.3. Hyperparameter Sweep and Model Architecture

In order to determine the most relevant parameters for our CNN architecture, we will run test different combinations of hyperparameters. Much of the information from this section came from the Habib and Qureshi 2018 research review [17], starting points for hyperparameter sweeping are from the Civilini et al. 2021 study [1]. Some parameters, listed below, we can test with the current starter architecture, so as to reduce the number of models while assessing feature effectiveness: Some parameters, listed below, we can test with the current starter architecture, so as to reduce

the number of models while assessing feature effectiveness: Some parameters, listed below, we can test with the current starter architecture, so as to reduce the number of models while assessing feature effectiveness:

- **Optimization Function (3 total):** The default optimization function is ADAM, in addition we will test the similar NADAM function, as well as SGD and the SGDF [18], the latter of which is discussed in the model architecture section.
- **Stride (2 total):** We will test stride lengths of 1 and 2. With our task, using a stride of 1 may be beneficial to pick up small differences between spectrograms.
- **Learning Rate (5 total):** We will test learning rates of 0.0001, 0.001, 0.1, and 0.05 to see which value is most effective.

After we determine the most effective of these preliminary hyperparameters, we can set these for the final model. For other hyperparameters, listed below, we will need to compare more incrementally, alongside other hyperparameters.

- **Number of Convolutional Layers (4 total):** In terms of the structuring of fully connected, convolution, and dropout layers, we will begin by structuring similarly to [1]. We will also model the number of layers after this study, testing 2, 3, 4, and 5 layer nets, to determine if neural nets with more layers are effective for this task, altering the stride length as needed.
- **Number of Epochs: (3 total)** Previous work has established 50 epochs to be sufficient for model learning. We will test 20, 40, and 60 epochs to see if this holds true.
- **Batch Size: (3 total)** We will test batch sizes of 8, 16, and 32, as discussed in Civilini et al. 2021.

After we determine the optimal configuration between these hyperparameters, we may adjust them further within smaller ranges, (i.e. if 40 epochs is ideal, we try 50 and see if this is better), to further enhance model learning. For instance, we will select 3 values between the two best performing learning models from the preliminary testing. Other parameters, such as the activation layer or specific pooling adjustments, can also be made at this stage. Furthermore, To reduce overfitting and optimize hyperparameters for the validation set, we used L2 regularization, which performed significantly bet- ter than early stopping. Additionally, we found that using the standard ReLU as our activation function and Mean Squared Error (MSE) as our loss function produced the best results. The preliminary testing parameters are changed one at a time, holding all others constant, meaning that we will need to run $3 + 2 + 5 = 10$ total trainings for the preliminary parameters. Then, we will need to run $4x3x3 = 36$ trainings for the main, iterative testing. Then the final round will be done on a case by case basis, and may result in 10-15 more training runs. We record the model accuracy and loss for each to determine which model is learning most effectively.

3.4. Classification of Unlabeled Data + Future Analyses

Once we determined a final architecture, we treated the unlabeled moonquakes as test data. Because there is high uncertainty in this classification due to the lack of established labels, we assign a confidence score to each moonquake based on Model 1 weights. We plan to send moonquakes with a confidence score of 0.99 or higher to expert geophysicists for official labeling.

4. Results

4.1. Input Spectrograms

To convert seismogram data to images, we generated spectrograms of one Grade A trace using 5 different 'nfft' values, 2 magnitudes on either side of the default (64, 128, 256, 512, 1024). For each of these 5 values, we set its 'noverlap' variable to be a fraction, namely (0.25, 0.5, 0.75) of the original 'nfft' value. The generated spectrograms can be seen in **Figure 2**. Furthermore, the 512 and 1024 NFFT tested on the Grade A data can be seen in **Figure 14** under Section 7, Supplemental Figures. From these images, we determined that an 'nfft' value of 256 combined with a 'noverlap' value of 128 yielded the crispest spectrogram (256,128), where noise was minimized and detail was maximized. The (128,64) spectrogram also proved effective, evidenced by the tighter banding which retained high clarity.

After analyzing the Grade A deep spectrograms, we performed a similar process with 3 other moonquake traces, being Grade A shallow, Grade A impact–to assess how spectrogram generation varied across moonquake type—-and Grade C deep, to see how waveform signatures varied across seismogram quality. The results of these generated spectrograms can be seen in **Figure 2**. One notable result is the clarity of the Grade C deep moonquake, meaning that parameter tuning for spectrogram translation may help to elucidate previously hidden patterns. Overall, the (128, 64) spectrogram produced the crispest result across the three seismograms, with (64, 32) also yielding an especially detailed result for the Grade A shallow. It seems as though (256, 128) results in color bleeding for events without as "powerful" as a marking, (i.e. shallow moonquakes), meaning that the (128, 64) spectrogram is the optimal choice to balance detail and crispness when formatting this

dataset across moonquake types and data qualities.

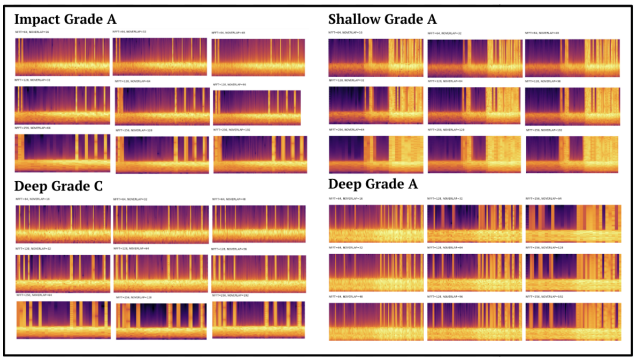

Figure 2. Shallow Grade A, Impact Grade A, and Deep Grade C Spectrograms

Apart from visual deduction, we further tested our model on a subset of grade A data using either (128, 64) or (256, 128) as part of our spectrogram generation functions. Through running the model with the already optimized hyperparameters, we concluded that the inputs consisting of (128, 64) spectrograms achieved a higher accuracy of over 5% higher than the same model with (256, 128) spectrograms as inputs.

4.2. Hyperparameter Sweep Phase 1:

To evaluate the performance of our moonquake classification model, our primary evaluation metric was classification accuracy and loss of the validation set. We determined our final model through completing a 2-phase hyperparameter sweep and selecting optimized parameters for the model through this process. Though, for the final paper we will evaluate the test set.

First, we compared the two main optimizers we are considering using for the final model, being SGD and Adam, and trained our model using both. We did not use the SGDF optimizer [18], as there are specific libraries we need to install beforehand. Overall, the Adam optimizer outperformed SGD significantly, reaching an accuracy of 70 percent as opposed to 15 percent for SGD, as seen in the Accuracy/Loss Results (**Table 2**). We will likely use Adam unless SGDF is extremely effective.

Next, using the Adam optimizer, we trained the model using a stride of 1 versus a stride of 2. We found that using a stride of 1 led to greater accuracy (70 percent vs 52 percent). The full accuracy and loss results can be seen in **Table 2**. It is worth noting that stride 1 is 5 times as computationally expensive on a per-step basis compared to stride 2, which may be impactful on our lagrer full dataset. If this is not problem, we will plan on using a stride size of 1.

Lastly, we tested the learning rate on the Adam optimizer. Originally, we had planned to test learning rates of (0.01, 0.05, 0.1, 0.2, 0.5), however, 0.01, 0.05, and 0.1 produced similar results of around 15 percent accuracy, so we tested learning rates of 0.001 and 0.0001, given that our initial estimates were too high. These produced an accuracy of 60 percent and 52 percent respectively and loss of 0.82 and 1.05 respectively, indicating that a learning rate near 0.001 would be most effective.

Overall, hyperparameter sweep phase 1 indicated that using the Adam optimizer, with a stride of 1, and a learning rate of 0.001 (or close to this), will be the most effective combination for the next phase of our hyperparameter sweep.

Hyperparameter Sweep Phase 1		
Parameter	**Accuracy**	**Loss**
Stride Length		
Length 1	0.6957	0.8821
Length 2	0.5217	0.9319
Learning Rate		
0.0001	0.5217	1.0528
0.001	0.6087	0.8187
0.01	0.1522	1.0961
0.05	0.1522	1.0918
Optimizer		
ADAM	0.7174	0.7807
Stochastic Gradient Descent	0.1522	1.0998

Table 2. Phase 1 Performance Metrics for Grade A Data. Parameters are organized by categories for clarity.

4.3. Hyperparameter Sweep Phase 2A:

After the completion of hyperparameter sweep phase 1, we tuned the number of layers, number of epochs, and batch size more iteratively using the optimized parameters from phase 1.

First, we varied the number of layers on the model, pairing 2- and 3- layer models with both a stride of 1 and 2, and pairing the 4- and 5- layer models only with a stride of 1 (stride of 2 was incompatible), similar to the layer setup in Civilini et al. 2021. We found that using 2 layers with a stride of 1 to be most effective, achieving an accuracy of 86 percent, similarly, 2 layers with stride 2 also performed well, (84 percent) with a steep drop-off in performance with the addition of more layers. The accuracy and loss can be seen in **Table 3**.

After this, we varied the number of epochs to be 20, 40, and 60, and we ran these on a general model consisting of 3 layers, similar to the baseline in hyperparameter sweep phase 1. The accuracy and loss results can be seen in **Table 3**. Overall, the model trained with 60 epochs produced the best results, with an accuracy of over 80 percent, as expected. Surprisingly, the model trained with 40 epochs yielded the lowest loss, (0.59), solidifying the notion that 50 layers may be an effective point to train at while minimizing loss [1].

Lastly, we varied the batch size to be 8, 16, and 32, similarly to Civilini et al. 2021, and found that the model trained with a batch size of 8 yielding the highest accuracy and lowest loss (**Table 3**). This is not surprising, as a smaller batch size allows for more detail to be considered by the model, yet is more computationally expensive.

Hyperparameter Sweep Phase 2		
Parameter	**Accuracy**	**Loss**
Number of Convolutional Layers - Stride Combination		
2 Layers - Length 1	0.8696	0.5196
2 Layers - Length 2	0.8478	0.6781
3 Layers - Length 1	0.5435	1.3312
3 Layers - Length 2	0.7391	0.7759
4 Layers - Length 1	0.7609	0.6537
5 Layers - Length 1	0.7174	0.6144
Number of Epochs		
20	0.5357	0.8855
40	0.7679	0.5926
60	0.8836	0.8422
Batch Size		
8	0.7321	0.5511
16	0.6724	0.8812
32	0.6167	1.1132

Table 3. Phase 2 Performance Metrics for Grade A Data. Parameters are organized by categories for clarity.

4.4. Hyperparameter Sweep Phase 2B:

After determining the optimizer, number of epochs, and batch size, we iteratively tested combinations of the number of convolutional layers and learning rates for the ADAM optimizer with all grades of data. We kept the same number of variations for the number of convolutional layers (2, 3, 4, and 5), but changed the variations for learning rate to be (0.0001, 0.005, and 0.001) based on the results of hyperparameter sweep phase 1. We also trained with a stride of 1 and 2 with 2 and 3 convolutional layers, with only a stride of 1 for 4 and 5 layers. Thus, the iterative hyperparameter sweep resulted in 18 trials to determine the final model. The results can be seen in Table 4.

Hyperparameter Sweep Phase 2		
Parameter	**Accuracy**	**Loss**
Model Configuration		
2 Layers - Length 1 - 0.0001	0.8871	0.3497
2 Layers - Length 2 - 0.0001	0.8387	0.3085
2 Layers - Length 1 - 0.005	0.7097	0.7618
2 Layers - Length 2 - 0.005	0.7097	0.7589
2 Layers - Length 1 - 0.001	0.9032	0.5220
2 Layers - Length 2 - 0.001	0.8871	0.3863
3 Layers - Length 1 - 0.0001	0.9032	0.3017
3 Layers - Length 2 - 0.0001	0.7258	0.9664
3 Layers - Length 1 - 0.005	0.7097	0.7613
3 Layers - Length 2 - 0.005	0.7097	0.7587
3 Layers - Length 1 - 0.001	0.9032	0.2907
3 Layers - Length 2 - 0.001	0.6935	2.2511
4 Layers - Length 1 - 0.0001	0.8871	0.3243
4 Layers - Length 1 - 0.005	0.7097	0.7643
4 Layers - Length 1 - 0.001	0.9194	0.2843
5 Layers - Length 1 - 0.0001	0.7903	0.8606
5 Layers - Length 1 - 0.005	0.7097	0.7577
5 Layers - Length 1 - 0.001	0.7097	0.7587

Table 4. Phase 2 Performance Metrics for Different Hyperparameters.

4.5. Final Model:

The two-phase hyperparameter sweep allowed our us to determine optimal parameters to train our model with. In the hyperparameter sweep, results were relative, as base models were used to keep computational expense low, and allow us to directly compare one set of parameters at a time. The model combines each parameter that produced the best result, meaning that the architecture consists of 2 layers with a stride of 1, 60 epochs, a batch size of 8, using the Adam optimizer set to a learning rate of 0.001 achieved the best result on Grade A Data only, henceforth referred to as model 1 [1], (Figure 3) and the 4 layer model with a stride of 1, 60 epochs, a batch size of 8, using the Adam optimizer set to a learning rate of 0.001 achieved slightly higher accuracy and extremely low loss, henceforth referred to as model 2 (Figure 4). Both models were trained on Grade A data only, as currently, spectrogram translation was not able to elucidate patterns within lower quality (Grade B and C) moonquake seismograms, and training results were not successful when grade B and grade C data were included. Both models reached 90% accuracy, outperforming previous models tested alongside them in the hyperparameter sweep, and demonstrating the effectiveness of this

[1] This model's accuracy and loss results do not exactly correspond to its entry in Table 2, as we held non-optimized parameters constant for testing within the same parameter (e.g. using 20 epochs across all layer-stride combinations), before running a final optimization

optimization method.

One encouraging result from the final model is the increased accuracy on deep moonquakes, accurately classifying all 12 in the dataset. Previously, our model struggled on deep moonquakes in particular. Shallow moonquakes are still not being classified well (2/4), but this dataset is very small compared to deep and impact, which may explain the result.

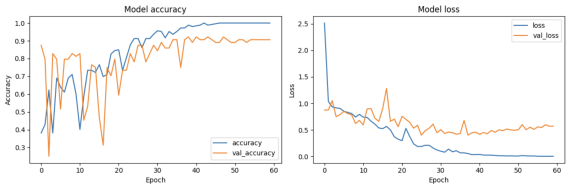

Figure 3. Model 1 Accuracy and Loss Results

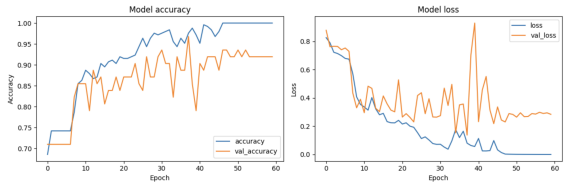

Figure 4. Model 2 Accuracy and Loss Results

Over all Grade-A PSE events in our training and validation sets, we were able to achieve over 90% validation accuracy and minimal loss. We compared our results and classification accuracy with prior studies using moonquake data to identify true moonquakes versus noise, true earthquakes versus noise [10], [1], source regions of deep moonquakes [8] [7], and lunar rockfalls [19]. Though classifying different processes of similar phenomena, these studies can serve as a threshold for the desired accuracy rates of classifications on lunar data. Our model achieved an overall accuracy of over 80%, outperforming previous studies (Majstorovic et al., 2024) [8]) by up to 5%. Our classification accuracy of deep moonquakes was particularly high, likely due to noise and fewer samples of shallow and impact moonquakes.

4.6. Final Model on Unclassified Data:

After training and optimizing our 2D CNN, we deploy our model to the classification of currently unclassified moonquakes. Currently unclassified moonquake recordings, also from the PSE and LSPE, were preprocessed following the same pipeline used for the training data, where each raw seismic signal was converted to a spectrogram. To do so, we employed Model 1, opting for this model despite slightly lower accuracy, from model 2,

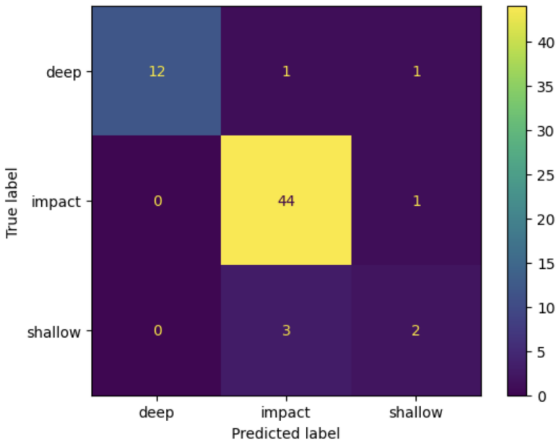

Figure 5. Final Optimized Model Confusion Matrix for Grade A Data

due to avoiding over-fitting. The entire process remained similar, but with the exception that only the forward pass of the process occurs, and the backward pass is skipped entirely. The result is a classification assigned the currently unclassified event (Figure 9).

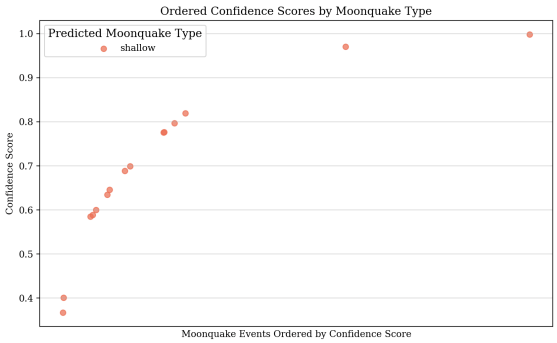

Figure 6. Confidence distribution over all predicted shallow moonquakes.

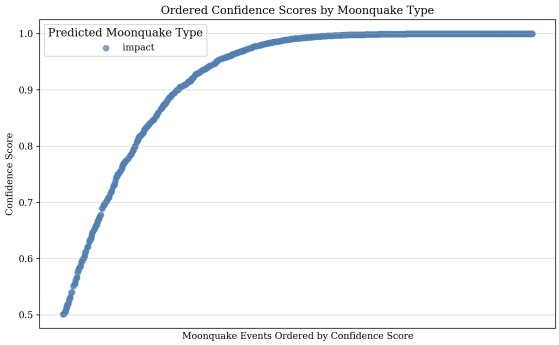

Figure 7. Confidence distribution over all predicted impact moonquakes.

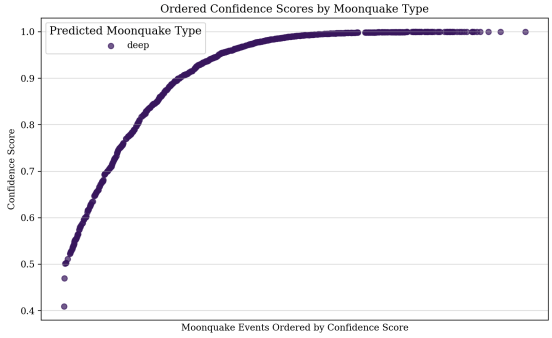

Figure 8. Confidence distribution over all predicted deep moonquakes.

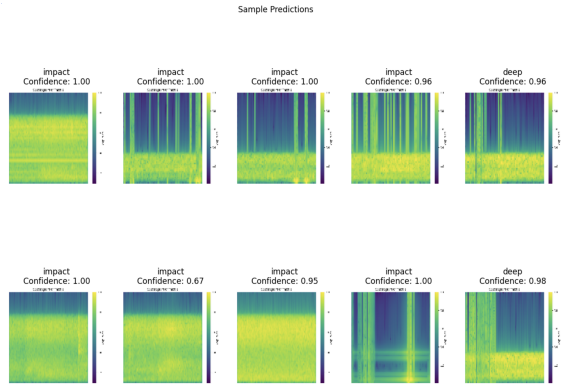

Figure 9. Example classifications assigned.

5. Discussion

This work and the evaluation of the best methods to detect and classify moonquakes has many important implications, including that funding for future seismic data may become more feasible due to the decreased need of sending back all collected data. Moreover, this further analysis of moonquakes provides better insight on the lunar structure and seismology of the moon. The machine learning methodology used in this paper may also be applied to earthquakes, potentially providing valuable assessments and early and automatic detection of earthquakes [16]. Perhaps most importantly, by improving CNN detection and classification of moonquakes, we could potentially improve accuracy and generalizability for real-time lunar seismic data. Additionally, we intend for this work to contribute to understanding how well classifiers can understand noise for future lander missions, such as on Mars, where "dirunal and seaonal" changes afftect data quality [3]. For future missions, automatic detection accuracy is essential for data storage, which is a scarce resource on lander missions and an area of focus for planned endeavors [1], a motivating idea central to the Civilini et al. 2021 paper. Furthermore, our methods may be able to label currently unclassified earthquakes into a defined category, which we aim to verify with an expert in the field, so unclassified moonquakes could be officially verified and labeled.

One major threat is including too many layers which overfit the data, especially when we are working with subsets of high-quality (Grade A) versus low-quality (Grade C) data. In order to combat this, we will simplify our model architecture and use a training set of varied quality. Another more fundamental threat to this research is that there is no ground truth for moonquake data, as the classification of moonquakes is ultimately subjective and determined by the hand-labeling of experts, and could fundamentally change as more data becomes available. For examples of this, see the Lunar section. Regardless, optimizing emerging machine learning techniques for lunar data represents a valuable area of research, that can be applied to current assumptions even if those assumptions change.

The model was successfully able to classify unclassified events at a high confidence rate, with over 1300 out of the 1473 in the training set being classified at over 0.9 confidence, as seen in Figure 10.

Additionally, all 3 moonquake types (shallow, deep, and impact) were classified with similar confidence levels, as exhibited by the confidence distributions separated by class (Figures 6, 7, and 15).

The distribution by type of the predictions varies significantly from the distribution found in our training data - as seen from Figures 11, 12, and 13, the predictions consist of a large majority of impact moonquakes, while the training set consists of more deep moonquakes. This discrepancy can be explained by several possible factors: first, the catalogue of currently unclassified moonquakes may simply contain a disproportionate amount of impact moonquakes, as deep moonquakes having been the most studied category and also with much more distinct and consistent features

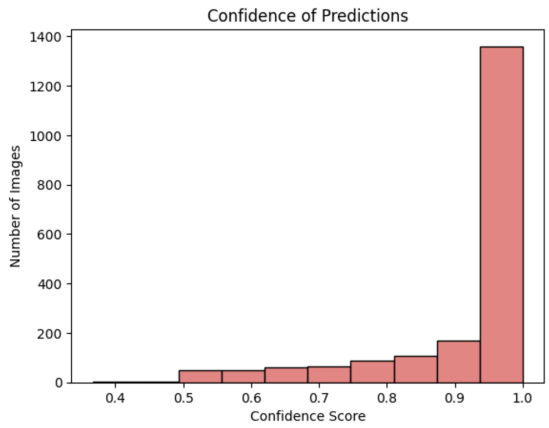

Figure 10. Confidence distribution over all unclassified predictions.

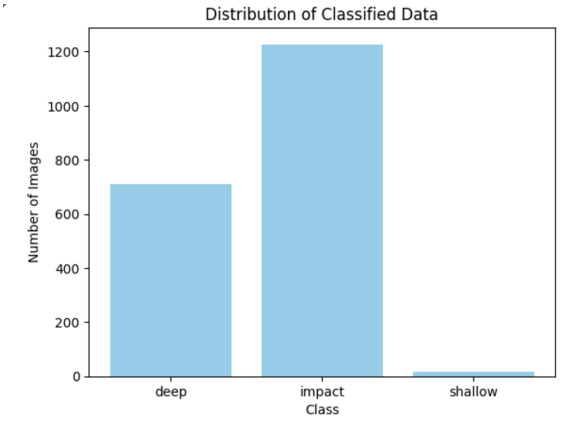

Figure 11. Distribution of predicted classifications

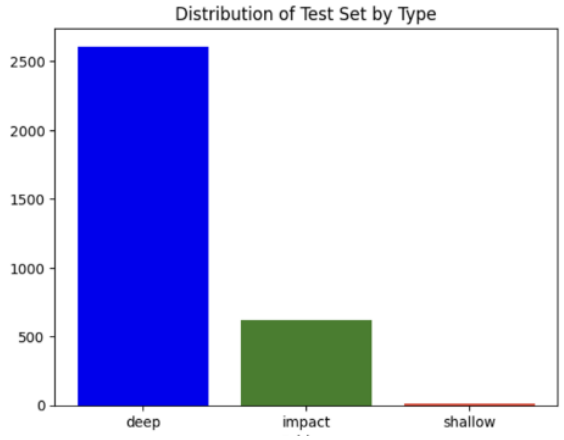

Figure 12. Distribution of predicted classifications

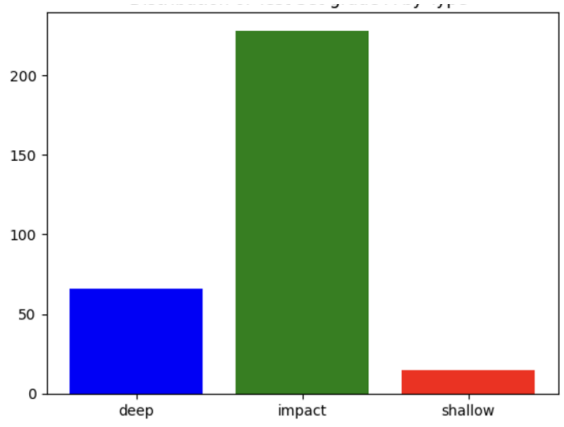

Figure 13. Distribution of predicted classifications

than impacts may have been classified to a higher degree and rate than impacts. Moreover, looking at the distribution by type when restricted to grade A data only (Figure 14), the distribution appears far more similar, with an even larger majority of impacts, indicating that most confident classifications done prior contain more impacts, despite the overall distribution containing more deep moonquakes. It may therefore be possible that some of the less confidently classified deep moonquakes are misclassified, leading to a real distribution that leans towards our predictions.

Because we have no holistic data for our test set, we look to verify our classification results with expert seismologists.

6. Conclusion

In this paper, we created a CNN classifier to label moonquakes into deep moonquakes, shallow moonquakes, or impact moonquakes. By adjusting the way we generate spectrograms from seismogram data, and performing a two-phase hyperparameter sweep, we created a model with over 90 percent accuracy on the Grade-A PSE events from the Apollo missions, and can be applied to unclassified data for potential classification and expert labeling. We believe this model is accurate and generalizable, and can help to assist experts in identifying currently unlabeled moonquakes, which has the potential to yield a more refined understanding of the lunar structure. Similarly to conclusions in Civilini et al. 2021, we believe that this model may help to develop methods to for future lander missions, or other applications where classification into waveform type is important for data prioritization and pattern recognition.

7. Supplemental Figures

References

[1] F. Civilini, R. C. Weber, Z. Jiang, D. Phillips, and W. D. Pan, “Detecting moonquakes using convolu-

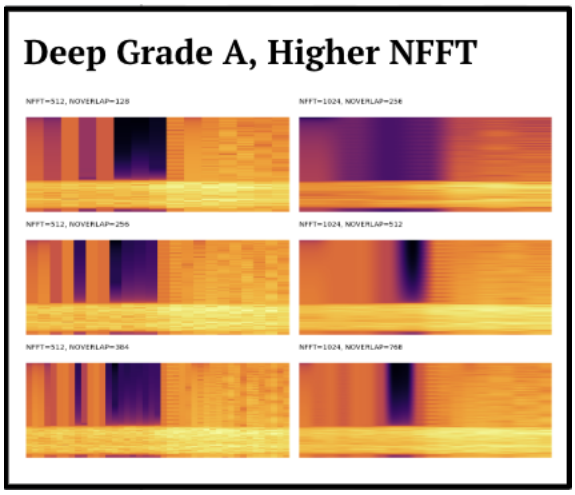

Figure 14. Example High-NFFT (512, 1024) Spectrograms with Grade A Data

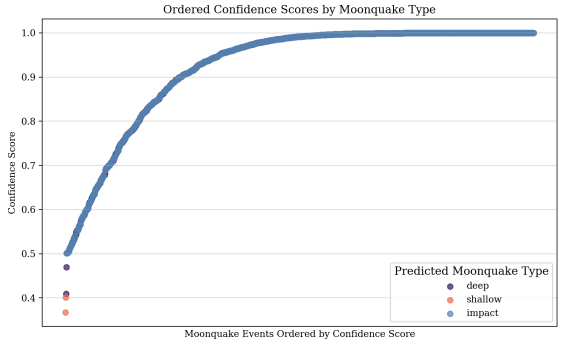

Figure 15. Confidence distribution over all predicted shallow, impact, and deep moonquakes.

tional neural networks, a non-local training set, and transfer learning," *Geophysical Journal International*, vol. 225, pp. 2120–2134, 03 2021. 1, 2, 3, 4, 6, 7, 8

[2] Y. Nakamura, "New identification of deep moonquakes in the apollo lunar seismic data," *Physics of the Earth and Planetary Interiors*, vol. 139, no. 3-4, pp. 197–205, 2003. 1, 2

[3] A. E. Stott, R. F. Garcia, A. Chédozeau, A. Spiga, N. Murdoch, B. Pinot, D. Mimoun, C. Charalambous, A. Horleston, S. D. King, *et al.*, "Machine learning and marsquakes: a tool to predict atmospheric-seismic noise for the nasa insight mission," *Geophysical Journal International*, vol. 233, no. 2, pp. 978–998, 2023. 1, 2, 8

[4] Y. Nakamura, G. V. Latham, and H. J. Dorman, "Apollo lunar seismic experiment—final summary," *Journal of Geophysical Research: Solid Earth*, vol. 87, no. S01, pp. A117–A123, 1982. 1, 2, 3

[5] R. Bulow, C. Johnson, and P. Shearer, "New events discovered in the apollo lunar seismic data," *Journal of Geophysical Research: Planets*, vol. 110, no. E10, 2005. 1, 2

[6] J. Majstorović, P. Lognonné, T. Kawamura, and M. P. Panning, "Identifying deep moonquake nests using machine learning model on single lunar station on the far side of the moon," *Authorea Preprints*, 2023. 1

[7] M. R. Schmerr C., Lekic. V, "Classifying deep moonquakes using machine learning algorithms," 1, 2, 7

[8] J. Majstorović, P. Lognonné, T. Kawamura, and M. P. Panning, "Predicting deep moonquake source regions using their temporal and spatial patterns and machine learning," *Journal of Geophysical Research: Machine Learning and Computation*, vol. 1, no. 2, p. e2023JH000117, 2024. 1, 2, 7

[9] W. R. Civilini F., Husker. A, "Thermal moonquakes in apollo seismic data," 2

[10] H. Dai and C. MacBeth, "Automatic picking of seismic arrivals in local earthquake data using an artificial neural network," *Geophysical journal international*, vol. 120, no. 3, pp. 758–774, 1995. 2, 7

[11] Z. E. Ross, M.-A. Meier, E. Hauksson, and T. H. Heaton, "Generalized seismic phase detection with deep learning," *Bulletin of the Seismological Society of America*, vol. 108, no. 5A, pp. 2894–2901, 2018. 2

[12] "Lunar seismology: A data and instrumentation review," *Space Science Reviews*, vol. 216, no. 5, p. 89, 2020. https://link.springer.com/content/pdf/10.1007/s11214-020-00709-3.pdf. 2

[13] K. Onodera, "New views of lunar seismicity brought by analysis of newly discovered moonquakes in apollo short-period seismic data," *Journal of Geophysical Research: Planets*, vol. 129, no. 7, p. e2023JE008153, 2024. e2023JE008153 2023JE008153. 2

[14] A. Khan, K. Mosegaard, and K. L. Rasmussen, "A new seismic velocity model for the moon from a monte carlo inversion of the apollo lunar seismic data," *Geophysical Research Letters*, vol. 27, no. 11, pp. 1591–1594, 2000. 2

[15] P. Lognonné, J. Gagnepain-Beyneix, and H. Chenet, "A new seismic model of the moon: implications for structure, thermal evolution and formation of the

moon," *Earth and Planetary Science Letters*, vol. 211, no. 1-2, pp. 27–44, 2003. 2

[16] L. M. Ho, J. I. Walter, S. E. Hansen, J. L. Sánchez-Roldán, and Z. Peng, "Evaluating automated seismic event detection approaches: An application to victoria land, east antarctica," *Journal of Geophysical Research: Machine Learning and Computation*, vol. 1, no. 3, p. e2024JH000185, 2024. e2024JH000185 2024JH000185. 2, 8

[17] G. Habib and S. Qureshi, "Optimization and acceleration of convolutional neural networks: A survey," *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 7, pp. 4244–4268, 2022. 3

[18] T. FILTER, "Signal processing meets sgd: From momentum to filter," 4, 5

[19] V. T. Bickel, C. Lanaras, A. Manconi, S. Loew, and U. Mall, "Automated detection of lunar rockfalls using a convolutional neural network," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 6, pp. 3501–3511, 2019. 7