D200s Computer Vision:
Technological Classification of Disaster Type and Damage Severity

Francine Bianca Oca (SID: 3039558030, franbianca.oca@berkeley.edu)
Dylan Do (SID: 3039556912 , _dylanddo@berkeley.edu)
Kassady Marasigan (SID: 3039560279 , kassady_marasigan@berkeley.edu)

Abstract

The application of combining both machine learning and data science abilities have made the innovative technological advancement of natural disaster recognition possible. By applying classification techniques through utilization of logistic regression as our mathematical basis, two different classification methods are possible on satellite imagery representing damage as a result of three types of natural disasters: Midwest Flooding, Socal Fire, and Hurricane Matthew. The first capability of our model is to classify a satellite image into one of two disaster types, flood and fire. The second classification capability of our model is to assign different levels of damage severity presented by the hurricane satellite images into the four ordinal categorical variables: 0, 1, 2, and 3 (0 exhibiting no damage and 3 equating to the complete destruction produced by the natural disaster). By training, validating, and testing our model based on the dataset adapted by xView2, our model for disaster type classification for Task A averaged a mean accuracy of **0.97349**, while the damage severity recognition for Task B produced an F1 score of **0.5056**. The dataset is one of the largest datasets that contain high-resolution satellite images of real-world building damage as a result of natural disasters. Pursuing this further, model training relied on the feature extraction of Sobel edge detection, Gabor filters, Local Binary Patterns (LBP), color intensity information, image size, pixel proportions, histogram of oriented gradients, contour count, segment count, and Euler number. These features were all used in different combinations for the model to become familiarized and understand different pixel patterns throughout the xView2 dataset. Though this innovation brings excitement in the realm of natural disaster rescue strategy planning, much is left to consider when fully depending on a model similar to this one we developed, such as possible model errors that may or may not interfere with the ability to recognize disaster type or damage severity, and the limitations of the dataset that was the foundational learning basis for the model to train on.

1. Introduction

Aside from the *physical* implementations of natural disaster response methods, there is no official computerized method that exists to replace the current dangers of rescue and recovery. According to Gupta et al. (2019), the technological familiarization of unique building damage would provide a more efficient practice in formulating the appropriate rescue approach, and this could possibly save a significant amount of time by allowing immediate action as opposed to spending a considerable amount of time determining the damage level produced by an unknown disaster type. Though there are many articles that discuss the benefits and challenges of utilizing artificial intelligence (AI), such as the limited access to high-quality datasets that represent a wide range of natural disasters and weather forecasts (Kuglitsch et al. 2022), the model that we developed is a great preliminary example that can be further improved and trained upon once more superior datasets become available.

The primary method of predicting the event of natural disasters can be mostly attributed to the use of satellites in a simple manner. With that being said, fully depending on a satellite's basic functionality for natural disaster predictions and classification can lead to significant problems, such as error or inability to analyze an image because of cloud coverage (Shukla et al., 2022). Moreover, another prominent concern in fully depending on satellites is the inconvenience of unwillingly needing to wait for the exact day and time that the satellite is in correct orbit above the location of interest (Shukla et al., 2022). This is inefficient for disaster strategy planning and requires the technological enhancement that our model aims to achieve.

With the problems of current satellite usage in mind, the goal of differentiating our model to improve ongoing practices in place involves higher image analysis sophistication. This sophistication is attributable from our implementation of features that will be further explored in the "Methodology" section of this report. By applying our version of feature extraction, the model we developed dives deeper into learning texture, color, size, and pixel patterns to investigate differences between the dataset images, thus learning unique configurations that classify each disaster type and damage level.

2. Description of Data Through Exploratory Data Analysis (EDA)

In the process of exploring our datasets, we loaded and visualized images from each disaster, which revealed that each raw satellite image had a corresponding label for its disaster type and damage level. We displayed three images from each disaster dataset to get a clear visual and gain further understanding of some basic statistics about the datasets we were working with. We created a visualization to plot the distribution of disaster labels amongst each disaster type as shown in the left image of Figure 1. This revealed that the majority of our images for Socal Fire and Midwest Flooding were labeled 0, while the damage labels for Hurricane Matthew were a little more spread out amongst the four labels. For this reason, the Hurricane Matthew dataset is well-suited for Task B which involves classification of *damage levels*, and the Socal Fire and Midwest Flooding datasets are better suited for Task A which involves the classification of *disaster type*. We then plotted the number of images contained in each disaster dataset as shown in the right image of Figure 1. This visual showed the distribution of images across the three disasters, highlighting a larger dataset for the hurricane compared to fire and flooding.

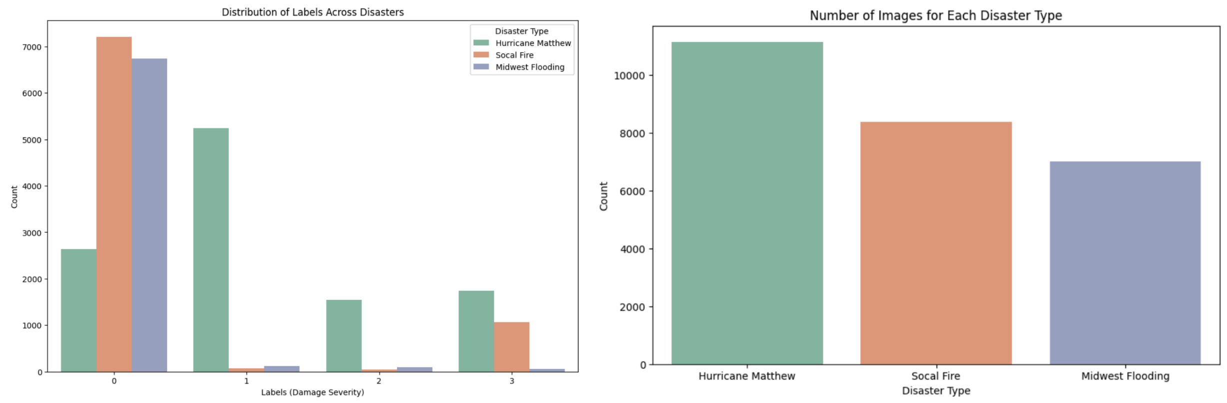


Figure 1. EDA visuals that show the distribution of damage labels amongst the three disasters (left) and the number of images per disaster dataset (right).

To further explore the data, we created violin plots as shown in Figure 2, to visualize the distribution of image sizes across the three disaster datasets. We made two visuals, one for the width dimensions and one for the height dimensions which showed a large variance in size, but similar distributions for each disaster. The interquartile ranges for each dataset, represented by the grey bars at the center of each violin, reveal that the majority of the sizes are under 200 in width and height, and larger images existed more so in the Socal Fire and Midwest Flooding datasets. Based on the violin plots, the variations in width and height would make for good features in the classification of disaster types since the distributions have a distinct spread per disaster type.

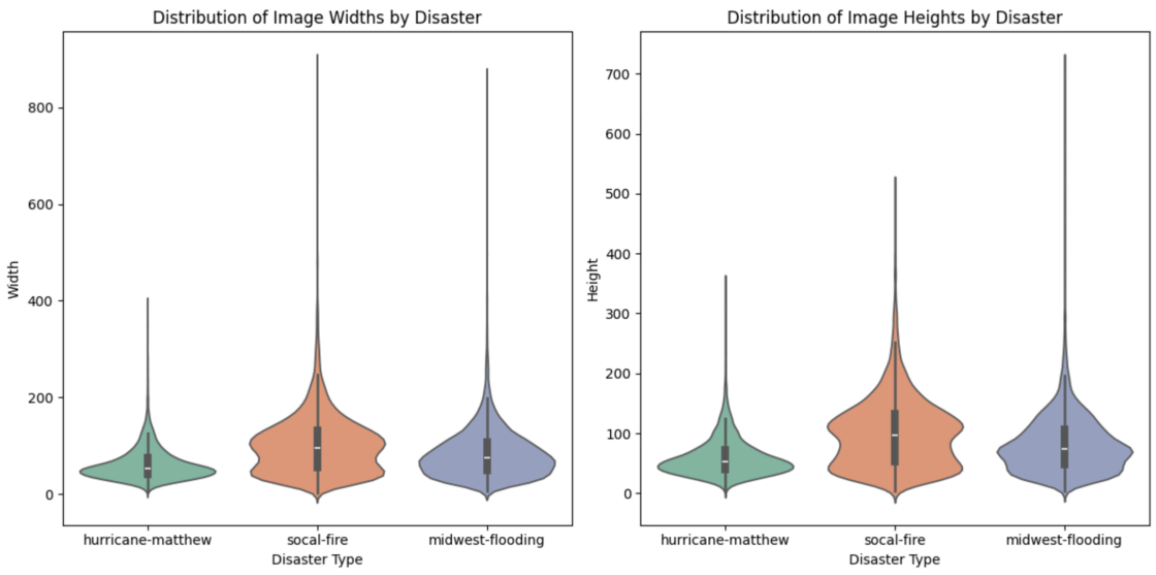


Figure 2. Violin plot that shows the distribution of image sizes by dimensions of width and height per disaster dataset.

The last visual we created to explore our data was a histogram, as shown in Figure 3, that effectively illustrates how the distribution of red, green, and blue (RGB) color intensities varies between images from the Hurricane Matthew, Midwest Flooding, and Socal Fire disaster types. This visual analysis aids in understanding which color features might help in distinguishing between different types of disasters necessary for Task A of this project and different damage levels necessary for Task B. The histograms reveal that while some color intensity overlap exists across disasters, there are distinct patterns unique to each type, and these differences in color distributions could be leveraged for effective classifications. Lastly, we created a function to ensure there were no missing or invalid data since including them into our model training may have led to a possible interference and/or inaccuracy in model learning, and they proved none.

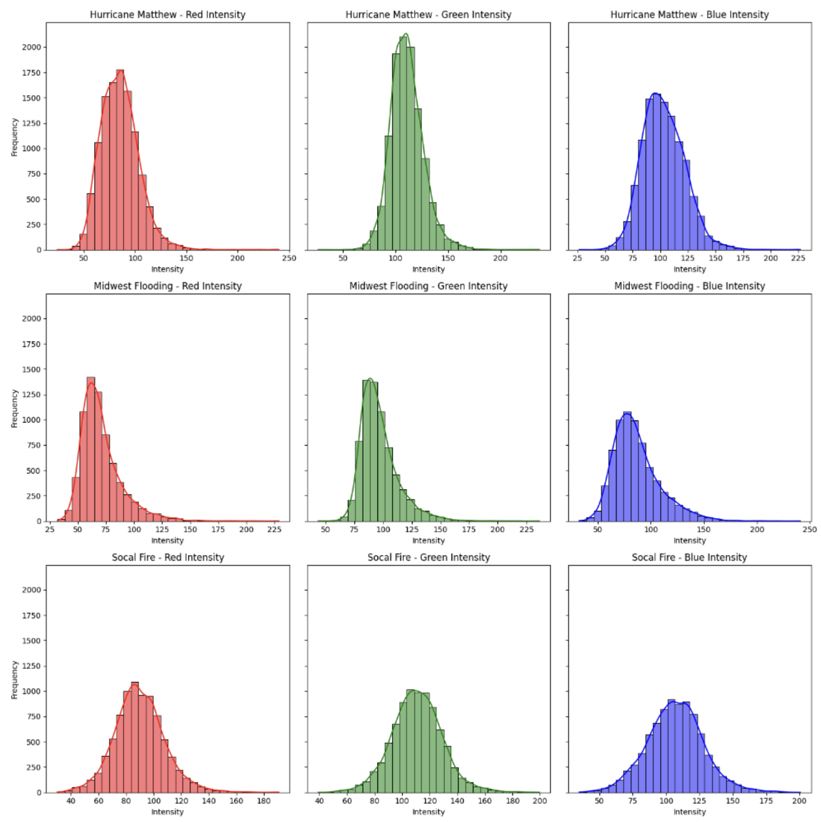


Figure 3. Histogram that shows the distribution of RGB color intensities across the three disaster datasets.

After much EDA, we moved on to feature engineering. The features we chose to extract from the datasets were derived from various image-based analyses such as Sobel edge detection, Local Binary Patterns (LBP), Gabor filters, color intensities measured across the RGB spectrum, image dimensions such as width and height, and black-to-white pixel proportions. Sobel edge detection is a gradient-based method that highlights edges in an image by looking for strong changes in its first derivative. Local Binary Patterns is a texture descriptor that thresholds the neighboring pixels of an image based on the value of the current pixel. Gabor filters are linear filters used to extract frequency features at multiple orientations. The RGB color channels give insight to color intensities of the images and the width and height dimensions were extracted from each image. In addition, for Task B, the black-to-white pixel proportions were used to measure the ratio of black to white pixels to give more insight to the texture and damage detection in each image. We included a visual of each feature in our Jupyter Notebook to see how each one transforms the image to provide valuable information that can be used in the training of our model.

After each feature was carefully selected, we moved on to data structuring. We chose to structure our data in a Pandas dataframe so that adding and extracting specific features would be easy and printing the dataframe would give us a clear visual of the numerical values held within it. After converting the image type, we were able to extract each feature for every image and append them to a single dataframe named ‘features_df’. The average values were taken for the features that outputted arrays like Sobel edge, LBP, Gabor filters, RGB intensities, etc.. After successfully combining our features, we were able to print the dataframe with each row representing an image from the xView2 dataset and columns containing its disaster type, all of the extracted features, and its corresponding damage label. With a functioning and well-prepared dataset, we were able to move onto the modeling phase.

3. Methodology

Model Development and Validation

The mathematical model that was implemented for the training logic of our AI model is logistic regression. The basis of choosing logistic regression as our foundation is due to the fact that it is widely used for binary classification problems. By using its probabilistic nature to train our model on the xView2 dataset, logistic regression was applied to the model's training process of classifying a satellite image and whether it represented damage as a result of Midwest Flooding or Socal Fire. Moreover, multinomial logistic regression was used to train the model for its capability to classify a satellite building image's damage level to either 0, 1, 2 or 3 (with 0 being no damage and 3 representing completely destroyed). The second task was strictly trained on the dataset containing Hurricane Matthew satellite images. The reason for applying multinomial logistic regression in the second task is because the damage severity labels are not limited to two discrete outcomes. To reiterate the importance of why we decided to emphasize the use of logistic regression for our model, it's noteworthy to include our reasoning as to why we didn't proceed model training with linear regression. Linear regression is well-suited for models that make predictions on data that have some sort of linearity. With that in mind, classification typically does not involve linear data and in fact, deals with non-linear discrete data. For the purposes of our model, we implemented the Python scikit-learn library to assign logistic regression as our model basis, in which we pursued further data fitting and training predictions. It is also noteworthy to mention that we applied sklearn's StandardScaler module to normalize our feature matrix, before performing cross validation. It should also be considerably mentioned that before training the model by using logistic regression, sklearn's train_test_split functionality was used to split the dataset into a training and validation set (in our case we used a test size of 0.2).

To validate the model prior to the test set of images, we proceeded with applying the K-Fold cross validation method. K-Fold validation involves the splitting of data into 'K' number of folds, where the model is then tested on those 'K' folds with unseen (or validation) data. A benefit in using K-Fold is the prevention of overfitting, where the model may be "overtrained" to the trained data, thus not predict and perform well on unseen data. The splitting of 'K' folds, allows the model to become familiarized with different types of data and allows it to become more robust, as it prevents the model from looking at the same data. To apply this in the development of our model, we used scikit-learn's KFold module, in which we defined K to be 5 for both classification tasks of assigning disaster types and damage severity.

Feature Extraction/Engineering

In regards to feature extraction, ten main features were extracted from the dataset for our model to train on: Sobel Edge Detection, Local Binary Patterns, Gabor Filters, and Color Information, Image Dimensions, Pixel Proportions, Histogram of Oriented Gradients, Count count, Segment count, and Euler Number. To simplify the process of extracting features, all features of all images were stored in a dataframe to be applied to the model and this allowed simultaneous consideration of all features we were interested in for our dataset and for *both* tasks. After completion of applying feature extraction into a dataframe, it resulted in having 26535 rows x 18 columns. The rows correspond to the 26535 images and the columns represent the 18 features. It is crucial to distinguish the disparity between the five main features explored in this section and the 18 features represented by the columns in the features dataframe. The reason why there are 18 features applied to each image in the dataset as seen in our feature_df dataframe is because some of those features consist of the averages of other features, to further assist with model learning as well as the disaster type and damage label, which are the features to be predicted.

The first feature of interest is Sobel Edge Detection and this feature allows the model to identify any possible gradient magnitude, which leads to the detection of directional change. In other words, it utilizes the grayscale difference between pixels to detect an edge (Tiar et al., 2021). The calculation of an image's gradient magnitude is crucial as it is the foundation of finding the intensity of a pixel. This foundation allows the model to see rate changes in light, and is useful in detecting different types of damage for different natural disasters, as well as the different levels of damage. In our study case, the function "get_sobel_features()" was applied to retrieve sobel information for each disaster image. This was considerably helpful for our model because it allows the model to determine what is considered damage as a result of natural disasters through the integrity of the edges that it detects.

The next feature used for model learning is Local Binary Patterns (LBP), which is a feature that has incredible discriminatory power while being quite conceptually simple. The way LBP operates is that it 'classifies' the pixels of an image depending on a certain threshold of the neighboring pixels. This leads to the result of binary classification of these pixels, thus allowing the representation of the texture of the image. The function named "get_local_binary_pattern()" was also applied to each image in the dataset to get that specific feature information. For the purpose of our model since we are concerned with disaster damage, LBP was useful in gathering texture details of our images in what may have possibly represented destruction.

Another feature that we implemented is Gabor Filters, which is similar to LBP, in which it studies the changes of texture in an image widely used for edge detection as well (Mehrotra et al., 1992). It operates in this way by calculating waves of energy on certain regions of an image, thus capturing patterns and textures where disaster damage may be present. Patterns that may be disrupted through cracks in an image represented by disaster damage can be indicated by a model, specifically by line irregularity that can be detected by gabor filtering. Similarly to the previous two features, the function called “get_gabor_features()” was also applied to each image in the dataset to extract gabor information.

Furthermore, the fourth feature that was explored for our model to learn patterns in the dataset was color information. By fetching the average RGB color intensities of each image, the model could possibly have learned color patterns in certain disaster damages as well as damage intensities. No specific function was created for this feature; However, the calculation of RGB intensities was done by accessing each image, and getting the mean intensity of each color channel. The values for the red color channel intensities could have played a strong factor in distinguishing the Socal Fire images while the blue color channel intensities could have done the same for the Midwest Flooding images. This method of using color intensities allowed the model to make more informed predictions by linking visual hues to specific disaster types.

The fifth main feature extracted was the dimensionality of the images, specifically in width and height. Our exploratory data analysis revealed that there was a variety of dimensions in the images from narrow slivers to broader captured views. Recognizing this variation as a potential indicator of the type of disaster, we included image dimensions as a feature in our model, hoping that similar disaster types could hold similar dimensionality for our model to train and predict on.

The sixth main feature was the black-to-white pixel proportions for each image. The reason this feature is important in terms of classifying damage level is that the dominant presence of black pixels can indicate ruined areas and debris that may indicate high level damaged areas. The function “calculate_black_white_ratio” was applied to images to extract the ratio. This feature is informative for the model to distinguish between each image’s disaster level in combination with our other features.

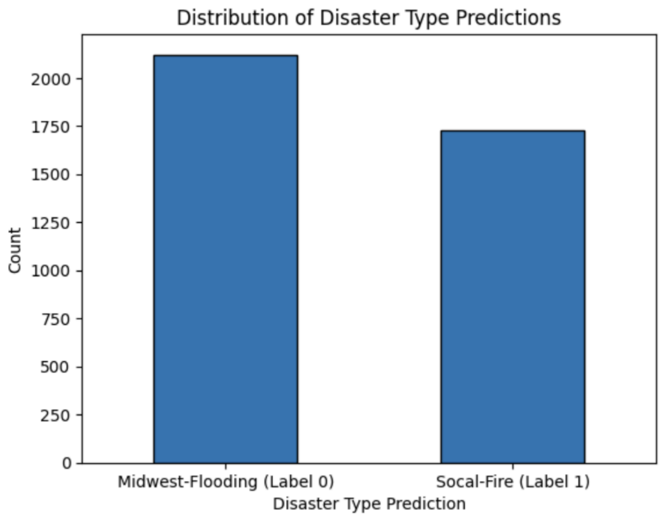
The seventh feature created was the histogram of oriented gradients (HOG). This is an skimage image feature that counts the number of times a gradient orientation occurs in specific areas of the image. This is highly effective in computer vision, as it is useful in object detection, high-quality local contrast, and detecting fine-scale gradients (Dalal et al., 2005). Using the scikit.feature.hog function helped determine HOG features and return the sum to input as a feature in the model.

Finding the number of contours and number of segments were used as our eighth and ninth features, respectively. Finding the number of contours involved using the skimage.measure.find_contours function to trace the regions of constant intensity within an image with a constant intensity value by using the marching square algorithm (Lorensen et al., 1987). Contour count is instrumental in analyzing the disaster images and assessing the extent of the level of severity. Using the SLIC segmentation algorithm from skimages, it divides the image into segments based on color and space, and the function will return the number of unique segments. This is important for disaster prediction as segmenting the different properties of the image can enable a more accurate assessment of the disaster.

The final feature implemented was the Euler number, which is the number of objects minus the number of holes (Stéfan van der Walt, et al., 2014). Calculating the Euler number is important because it can provide information about the topology of the image, for example, it can distinguish the complexity of features of the image such as infrastructure and bodies of water. This is helpful in indicating the severity level of each disaster as well as distinguishing different disaster types.

4. Summary of Results

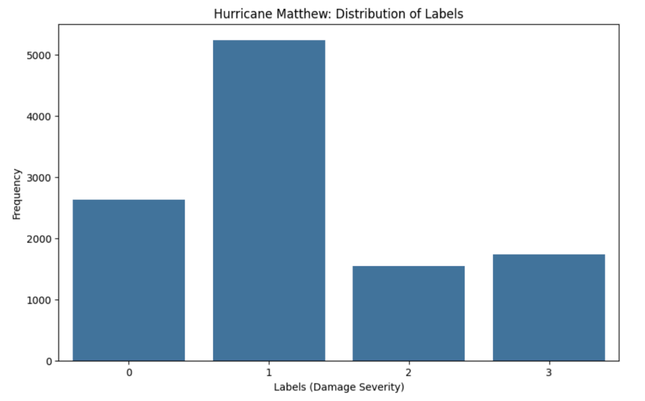
In Task A, our objective was to classify images based on their corresponding disaster type between the Midwest Flooding and Socal Fire. Initially, our approach to structuring the dataset involved using NumPy arrays, which yielded a low classification accuracy of about 0.54 in the training/validation stage. We added a large amount of features, but the accuracy of our model was not increasing which prompted a strategic shift to using pandas dataframes instead. We employed logistic regression which yielded a notable validation accuracy of 0.96425 in the training stage and 0.97453 in the testing stage via submission to Gradescope. This significant improvement could be due to the advanced data manipulation capabilities that pandas has. Dataframes provide more flexibility for feature engineering, which allows us to better identify and incorporate the right features into our model such as image attributes that were previously difficult to isolate using the NumPy array format. The training and test accuracies indicated a well-fitting model as the logistic regression managed to handle the variance of the dataset without overfitting. These promising results could be attributed to the choice and processing of features, which helped the model to better distinguish the texture, edge, color, and size differences between flooding and fire images to then classify them with high accuracy. Figure 4 below shows the spread of images predicted to be Midwest Flooding versus Socal Fire.



```
Count for Midwest-Flooding (Label 0): 2121
Count for Socal-Fire (Label 1): 1726
```

Figure 4. Distribution of predicted disaster types from Task A's logistic regression model.

In Task B, the objective was to create a damage level classifier, specifically for the Hurricane Matthew disaster. Just like Task A, we initially made NumPy arrays to predict our model, but yielded poor results, with a classification accuracy of 0.36 for the training/validation. However, after changing and modifying the data to be structured as a data frame and fine-tuning the features, the validation accuracy increased to 0.557. In addition, the F1 score of our training was 0.52 and predictions for testing was 0.5079 via submission to Gradescope. The F1 score measures the model's accuracy by calculating the mean of precision (number of correct positives divided by number of all positives) and recall (number of correct positives divided by number of positives that should have been correctly identified). This score means that our model for Task B is not perfect, but better than random. This suggests that more improvements to the model still have to be made in order for the model to be more reliable. Unlike Task A, calculating the black to white pixel proportions was an additional feature added to increase the validity accuracy. In addition, height and width were not taken into consideration as features for Task B because it led to an imbalanced dataset for the model to predict and resulted in lower accuracy. In Figure 5 below, the right graph is the distribution of labels for Hurricane Matthew in our original training dataset. The left plot showcases our test predictions for label distribution for Hurricane Matthew. Both plots show similar distributions where label 1 (minimal damage) is the most frequent classification. Next is label 0, where no damage has occurred, as the next highest frequency. Following this is label 3 indicating the most severe damage being more common than label 2. These results suggest that the model captured the underlying trend in damage severity for Hurricane Matthew. However, fine-tuning the logistic regression model parameters and adding more features can improve its sensitivity to varying levels of severity and enhance the accuracy of the validation.



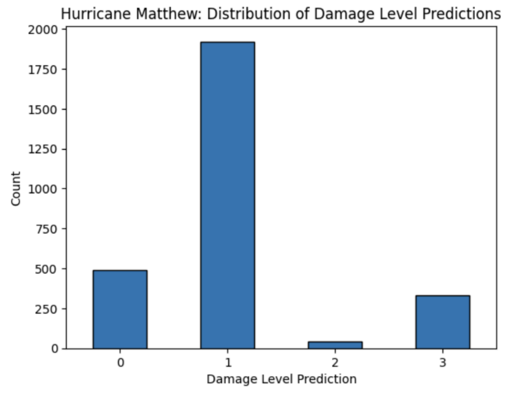


Figure 5. Distribution of labels for Hurricane Matthew. Right bar plot represents training images label distribution. Left bar plot represents disaster type distribution predictions from Task B's logistic regression model.

5. Discussion of Societal Impacts and Ethical Concerns

Although our model has been proven to perform well in terms of detection capability with both tasks of disaster type classification and damage level severity recognition, there still remains much room for model improvement for the following reasons.

Despite the usefulness of utilizing sobel edge detection as a feature, it's important to note that it comes with the limitation of its possible ineffectiveness in working with 'noisy' images. Since this feature relies on the differentiation between gray pixel values to indicate an edge, this detection method has a poor 'anti-noise performance' in the event of a noisy image - this is because changes in the gray pixel area become obscured in the presence of noise (Wenshuo et al., 2010). Pursuing model improvement further, unique adjustments to sobel edge detection may show potential in better model performance if certain changes are applied to make detection very specific for our purposes. For example, de-noising techniques such as image pixel value transformations may be applied to handle noise - this may include the development of a mathematical function to apply to pixel values which can be changed and tested to best suit our model (Wenshuo et al., 2010).

Another noteworthy limitation to consider in our model is the lack of ability of gabor filtering and LBP to analyze textural components of a low-resolution image. In the case of low resolution images, gabor filtering and LBP may have the negative effect of misclassification due to inaccurate textural pattern analysis. The xView2 dataset used to train our model contained relatively low-resolution and considerably pixelated images, so that is an important factor to consider when analyzing the accuracy results of our model as well.

Additionally, during model training, our runtime took a considerably long amount of time and this may be due to the model trying to incorporate a combination of those features. In other words, the model may also be computationally expensive, and even more so when working with a large dataset such as xView2. This was considerably true for the case when we were originally working with NumPy arrays because they were not well-suited for handling datasets with diverse data types or complex data manipulation like filtering, merging, applying functions, etc. These operations, we realized, could be computationally expensive and could increase the runtime of our model substantially. After switching to pandas dataframes, we observed a huge reduction in processing time to about a minute which was a great shock to us. This improvement is due to panda's optimized performance for data manipulation which was clearly observed in the case of our feature engineering and model taks. The employment of acknowledging and taking steps to resolve these limitations may have the possibility to improve our model performance.

A surprising discovery we encountered was in the process of attempting to improve model accuracy through use of a correlation matrix heatmap. We implemented a correlation matrix as shown in Figure 6 below, to evaluate the relationship between the various features and its disaster labels. Based on the low correlation between the lbp_mean, width, and height features with the disaster_label, we decided to remove them from the model to see if the performance improves. To our surprise, the training accuracy went down to about 0.855 for Task A. In addition, when taking out the color means in Task B due to negative correlation to the labels, the training accuracy decreased to 0.507. This unexpected outcome made us realize that features with seemingly weak individual correlations could still contribute significantly to the predictive label, possibly through interactions with other features. In addition, many other features were added and tested such as taking the rank mean and boundary mean from the images. These did not improve the accuracy of the model, suggesting that the more features that are added, does not necessarily equal the model's improvement. This discovery highlighted the importance of validation testing when modifying features based on correlation matrices.

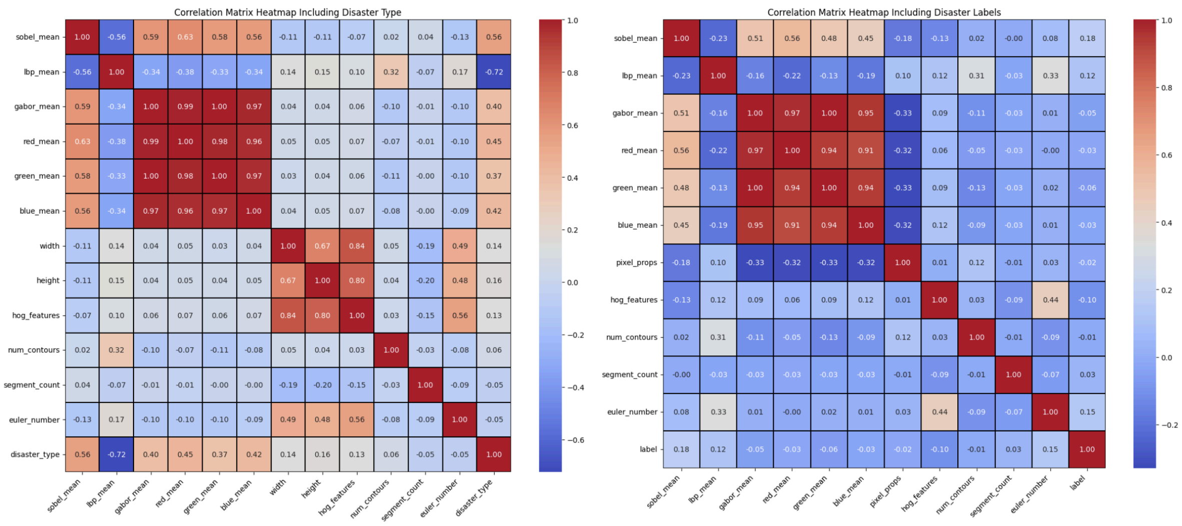


Figure 6. Correlation matrix heatmap of the features and labels used for predicting tasks. Left heat map represents features and labels for Task A. Right heat map represents features and labels for Task B.

6. Conclusion

In conclusion, our model resulted in moderate performance based on how well it was able to predict disaster types as well as damage severity. From Task A, the average training accuracy was 0.96425 and the testing accuracy was 0.97453 (of which both are relatively close and demonstrate good performance). With that being said, it is conclusive that the model has great potential in predicting the type of natural disaster that is depicted based on the satellite image presented. Although the training and testing accuracy is relatively high, it would be an interesting extension to this study to explore additional features that may or may not possibly improve the mean accuracies presented by the model. The features that are contributable to this accuracy are the following: 'sobel_mean', 'lbp_mean', 'gabor_mean', 'red_mean', 'green_mean', 'blue_mean', 'width', 'height', 'hog_features', 'num_contours', 'segment_count', and 'euler_number'. In regards to Task B, the F1-score based on training data was 0.52 and the score based on testing data was 0.5079. It's important to note that the features utilized to train Task B are the exact same as Task A except instead of 'width' and 'height', Task B applied 'pixel_props' as a feature.

For further exploration, the model is well suited for Task A in which the resulting accuracy deems so. However, to further improve Task A, it would be interesting to inspect other new features that our model has not investigated, to learn more about model sensitivity to certain features amongst others. The F1-score for Task B suggests that advancement is possible and recommended, maybe in testing different combinations of hyperparameters or even exploring a different modeling approach. As the F1-score is representative of the balance of precision and recall, a score of 0.5079 generally is defined as “moderately” balanced, but should definitely be augmented.

Though the model does not have perfect predictions, our study shows wonderful and insightful preliminary results that give way for additional research and testing to be done. These results alone are beneficial in heading towards the right direction to improving current practices of natural disaster rescue and recovery practices. More importantly, this research conducted by our group are great first steps in attempting to change the way strategic rescue strategies are done from physical implementations to a more innovative computerized operation that is both timely and efficient.

Link to video recording: https://youtu.be/ocfanSJfawM

References

Gupta, Ritwik, et al. "XBD: A Dataset for Assessing Building Damage from Satellite Imagery." *arXiv.Org*, 21 Nov. 2019, arxiv.org/abs/1911.09296.

Kuglitsch, Monique M., et al. "Facilitating adoption of AI in Natural Disaster Management through collaboration." *Nature Communications*, vol. 13, no. 1, 24 Mar. 2022, https://doi.org/10.1038/s41467-022-29285-6.

Tian R, Sun G, Liu X, Zheng B. Sobel Edge Detection Based on Weighted Nuclear Norm Minimization Image Denoising. *Electronics*. 2021; 10(6):655. https://doi.org/10.3390/electronics10060655.

Mehrotra, R., et al. "Gabor filter-based Edge Detection." *Pattern Recognition*, vol. 25, no. 12, Dec. 1992, pp. 1479–1494, https://doi.org/10.1016/0031-3203(92)90121-x.

Wenshuo Gao, et al. "An improved Sobel edge detection." *2010 3rd International Conference on Computer Science and Information Technology*, July 2010, https://doi.org/10.1109/iccsit.2010.5563693.

Shukla, Kumar A., et al. "DL based system for on-board image classification in real time, applied to disaster mitigation." *2022 Seventh International Conference on Parallel, Distributed and Grid Computing (PDGC)*, 25 Nov. 2022, https://doi.org/10.1109/pdgc56933.2022.10053139.

Dalal, N. and Triggs, B., "Histograms of Oriented Gradients for Human Detection," IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005, San Diego, CA, USA. https://ieeexplore.ieee.org/document/1467360.

Lorensen, William and Harvey E. Cline. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. Computer Graphics SIGGRAPH 87 Proceedings) 21(4) July 1987,p.163-170).https://www.researchgate.net/publication/202232897_Marching_Cubes_A_High_Resolution_3D_Surface_Construction_Algorithm

Stéfan van der Walt, Johannes L. Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D. Warner, Neil Yager, Emmanuelle Gouillart, Tony Yu and the scikit-image contributors. scikit-image: Image processing in Python. PeerJ 2:e453 (2014) https://doi.org/10.7717/peerj.453