Natural Disaster Classifier: Predicting natural disaster type and damage levels through linear support vector machines and random forest classifiers

Data 200S Graduate Project

Jared Paul Guevara, Laura Jones, Brandon Robello

Spring 2024

1 Abstract

The following report details the development of two predictive classifier models, capable of determining the type and extent of damage of certain natural disasters, using a set of pre- and post-disaster images from Hurricane Matthew, Midwest US floods, and Southern California (SoCal) wildfires. Image data was examined as multi-dimensional matrices, with features being visualized as value distributions and extracted from physical, color, and texture information.

For each task, two classifiers were utilized and compared against each other: the linear support vector classifier (LinearSVC) and the random forest classifier (RFC). Features were filtered by training each model on a subset of feature combinations and returning the subset of features that returned the highest training score. Optimal hyperparameters were determined by performing grid searches on each model. The RFC model performed better in both tasks, capable of differentiating between the Midwest flooding and SoCal fires images with a cross-validation score of .985 *across 10 folds. Additionally, the RFC model achieved a higher average F1 score of* .626 *when predicting the damage level of Hurricane Matthew images, on a scale from 0 (no damage) to 3 (destroyed).*

The project proved to be a valuable learning experience in creating a successful image classifier model without the use of a convolutional neural network (CNN), backed by rigorous exploratory data analysis (EDA) and feature transformations. The project video recording can be found here: `https://www.youtube.com/watch?v=zJm1dGayTyc`.

2 Introduction

Computer vision tasks seek to emulate and automate the abilities of the human visual system through the use of artificial intelligence and machine learning (AI/ML). The ever-growing field of computer vision has many applications that span far and wide, such as medical diagnoses and self-driving vehicles. The motivation for this project stems from the xView2 Challenge, a publicly available dataset of images from various natural disasters. The challenge is to leverage computer vision to automate the assessment of post-disaster building damages.

The dataset is accompanied by a descriptive article from the Defense Innovation Unit (DIU), detailing how the images were collected, annotated, and analyzed. Much of the focus of the article was dedicated to evaluating building damage, emphasized by the annotated polygons representing structures in each image. The article also features its own baseline ResNet50 model to assess building damage, on a scale from ”0 - No damage”, ”1 - Minor damage”, ”2 - Major damage”, and ”3 - Destroyed”. The baseline model achieved a weighted F1 of 0.2654, struggling particularly with classifying level 2 images as level 1 due to class imbalance.

The ResNet50 model is a specific type of CNN, which is the most up-to-date and effective model for computer vision and understanding images from an algorithm feature space. However, the following analysis is more deliberate and extracts features from the dataset directly for the training of both LinearSVC and RFC models. This examination, while not located at the forefront of image processing research, is an exercise that improved our collective understanding of the intricacies of computer vision and exemplifies that more traditional machine learning techniques are still effective and efficient at solving complex problems.

3 Description of Data

3.1 General Characteristics

The provided dataset contained a total of 26,535 images, with 11,151 images of Hurricane Matthew, 7,004 images of the Midwest flooding, and 8,380 images of SoCal fires. Each image was accompanied by a singular label, indicating the damage level. Utility functions were provided to extract images and labels, as well as apply image processing filters for feature extraction. No missing or null images were encountered.

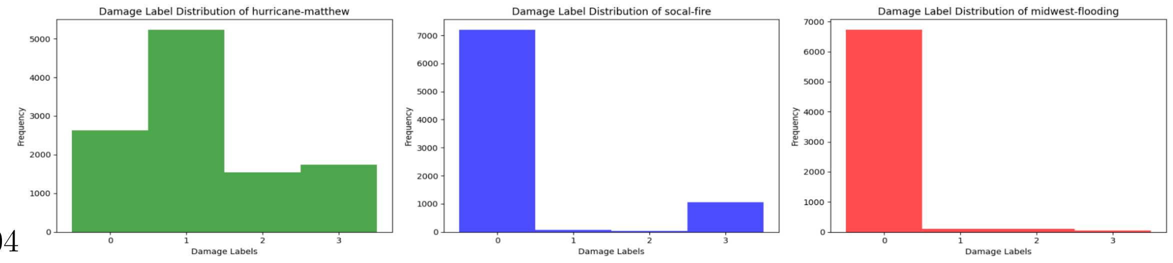

Figure 1: Distribution of damage labels across 3 disaster types

3.2 EDA

Initial EDA consisted of visualizing height, width, and image size distributions (see supplemental materials pg.9), as well as color distributions among the disasters (see supplemental materials pg.10). Group-by operations were performed to find potential differences and features of interest between the SoCal fires and Midwest flooding images, in addition to differences between damage levels from the Hurricane Matthew images. On average, the images from the Midwest flooding tended to be smaller than those from the SoCal fires, whereas level 3 images from the Hurricane Matthew set tended to be smaller than the other damage levels. Additionally, the fire images had greater color channel intensities (0.35 for red, 0.44 for green, and 0.43 for blue) than the flooding images (0.28 for red, 0.38 for green, and 0.34 for red). This difference made color channel intensity a feature candidate for categorizing images as either flooding or fire instances for Task A. The Hurricane Matthew images, however, did not show any discernible difference in average pixel intensity (see summary in Supplemental Materials, pg. 11).

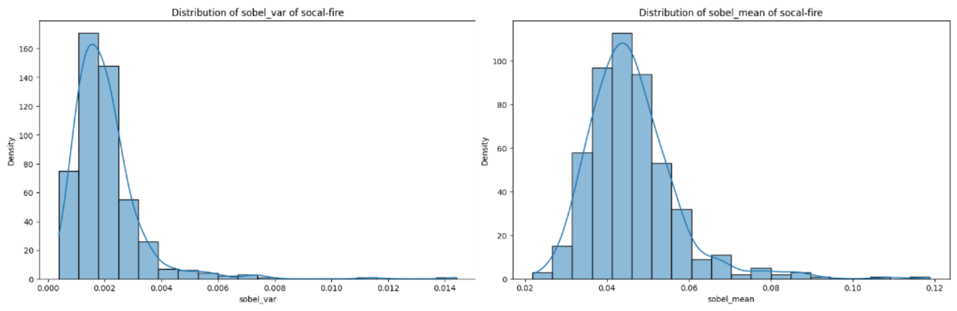

Figure 2: Distribution of Sobel Mean and Variance

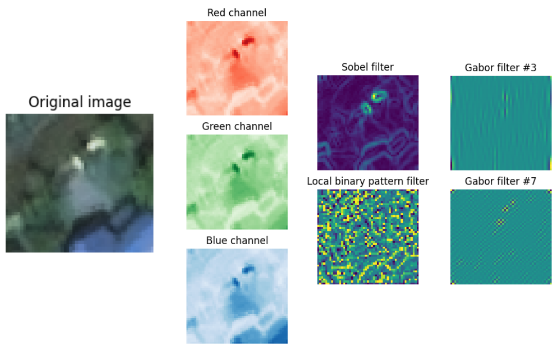

Figure 3: Kernel Filter Examples

More features were gathered from the original image matrices by applying the Sobel, Gabor, and Local Binary Pattern (LBP) filters (Fig. 2 & 3). The Sobel filter is an important kernel for edge detection as it "emphasizes regions of high spatial frequency that correspond to edges" [Fis+03]. Gabor and LBP filters are often used for texture analysis as model features [Tra98; Pie10], which are relevant to this project for differentiating between structures and topography in our image dataset. Rigorous work went into extracting these features, as twelve Gabor kernels, along with the Sobel and LBP filters, were applied to calculate the mean pixel intensity and variance of each image in each disaster. The resulting data was collected in a feature data frame, which was then split by task. Each task data frame was then trained on a standalone RFC to isolate which features were most important in predicting the correct labels via sklearn's ClassificationReport. Features with relatively large importance values were then saved to be appended to the final task data frames.

To create the task data frames for modeling, the first features added were those from the raw, provided images. Dimensional features such as height, width, and image size were added, along with aspect ratio and average pixel intensity across each color channel. Image-processing features from the previous step were then appended, followed by the last group of features, consisting of the percent color coverage of each image. A custom function was developed to calculate the ratio of pixels in an image that cross a certain intensity threshold over the total area. These thresholds were calculated as the average of pixel intensities across all labels in a given task (Fig. 4).

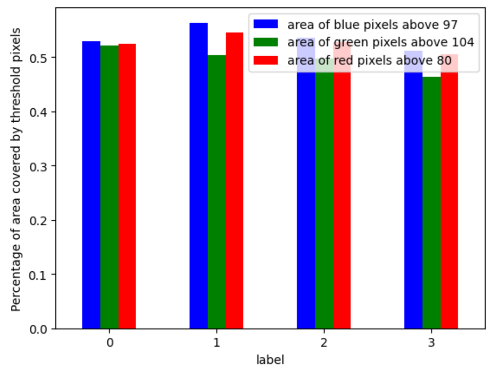

Figure 4: Comparison of color channels meeting threshold

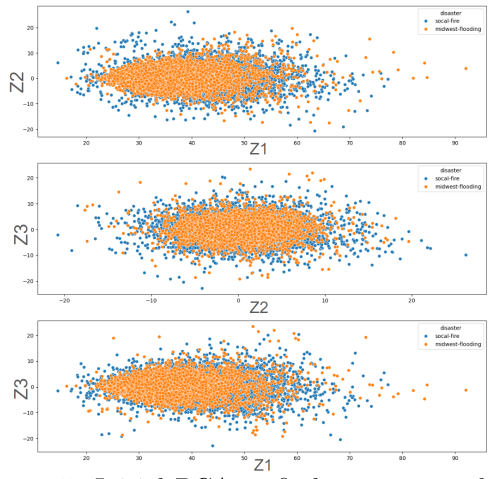

Figure 5: Initial PCA to find component clustering among disaster types

Principal Component Analysis (PCA) was performed using sklearn's Decomposition module, after scaling Task A's feature data. However, the analysis technique revealed little in the way of differentiating between SoCal fire and Midwest flooding disasters, with the principal components intermingling with one another (Fig. 5). Because of this, PCA was not performed on Task B's feature data.

4 Methods

The preceding EDA and following modeling were done using both Google Colab and Github as a means of collaboration and version control. LinearSVC and RFC models were chosen due to their aptitude for performing binary and multi-class classification on smaller, labeled datasets (less than 100K samples).

4.1 Class Imbalance

Prior to modeling, one issue that had to be addressed was the class imbalance among damage labels. As seen in figure 1, the majority of images in Hurricane Matthew are label 1, while the majority of both Midwest flooding and SoCal fires are label 0. For Task A, damage labels were not re-balanced due to both sets having a roughly equal numbers of samples as well as a classification goal aside from damage labelling.

For Task B, however, the classification of damage labels called for class balancing to aid in the model-training step. Training a model on the initial distribution of images would lead to similar issues from the xView2 article, where certain images were not classified properly due class imbalance.

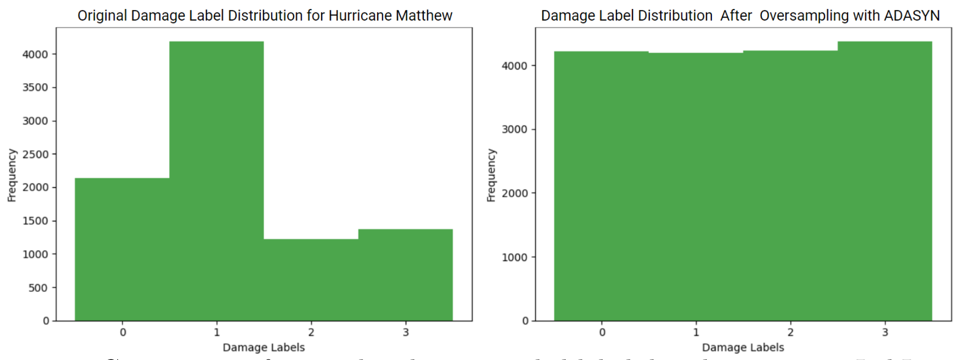

Figure 6: Comparison of original and oversampled label distribution using ImbLearn

To alleviate this problem, three different techniques were used to re-balance the labels: Random Undersampling (RUS), Random Oversampling, and Adaptive Synthetic Oversampling (ADASYN). The features for Task B were resampled using each technique, followed by training a standalone RFC to extract the weighted F1 scores, similar to the feature extraction step during EDA (results summarized in Fig. 7).

Of the three techniques, ADASYN had the highest F1 score of 0.57, while also generating the most samples amongst all three techniques. The following modeling steps for Task B utilize the resampled data from ADASYN, for a total of 21,229 samples.

Figure 7: Oversampling technique and constituent F1 scores and damage label quantities

Technique	**F1 score**	**0**	**1**	**2**	**3**
No resampling	n/a	2138	4190	1217	1375
RUS	0.52	1217	1217	1217	1217
ROS	0.55	4190	4190	4190	4190
ADASYN	0.57	4214	4190	4226	4377

4.2 Feature Selection

To select the optimal set of features and prevent collinearity, the entire feature space was split into three categories: raw features (i.e. height, width, average pixel intensity), image processing features (i.e. Sobel, Gabor, LBP means and variances), and color coverage features. The data was then allocated to test and validation splits, creating training sets to extrapolate scores from. Each combination of features was then used to train either a LinearSVC or RFC model and return the training score. Doing so revealed that, for both models and both tasks, the highest training score and optimal set of features was associated with both the image-processing features and color coverage features, as shown below.

Task A Features (both models)	Task B Features (both models)	
lbp_var	lbp_var	gabor_9_var
lbp_mean	gabor_7_var	gabor_11_var
sobel_mean	gabor_6_var	gabor_3_var
gabor_7_var	lbp_mean	gabor_1_var
gabor_3_var	sobel_mean	gabor_10_var
area of red pixels above 80	gabor_5_var	gabor_2_var
area of green pixels above 104	gabor_4_var	gabor_8_var
area of blue pixels above 97	sobel_var	gabor_0_var
	area of red pixels above 85	
	area of green pixels above 111	
	area of blue pixels above 102	

4.3 Hyperparameter Tuning

To tune the hyperparameters of each model, grid searches were performed using sklearn's GridSearchCV, which takes in a set of specified hyperparameter options to return the combination of hyperparameters with the best score. For the LinearSVC models, the parameter grid consisted of 'C' (regularization parameter), max iterations, and the penalty type. For the RFC models, the parameter grid consisted of the max depth, max number of features, and number of estimators (aka the number of trees). The optimal parameters for each model and task are shown below.

Task A LinearSVC hyperparameters	**Task A RFC hyperparameters**

C = 10
max iterations = 3000
penalty = L2
	max depth = 10
	max features = log2
	number of estimators = 1000

Task B LinearSVC hyperparameters	**Task B RFC hyperparameters**

C = 10
max iterations = 3000
penalty = L2
	max depth = 10
	max features = 0.33
	number of estimators = 100

4.4 Model Training

The modeling pipeline for both tasks involved splitting the data into training and validation sets, scaling, fitting the data, and generating accuracy scores for both splits, along with a cross-validation score across 10 folds. Additionally, the predictions for each set were saved to generate confusion matrices and classification reports of each model's performance (See Supplamental Materials pg. 13).

5 Summary of Results

5.1 Task A

Training, validation, and cross-validation scores for differentiating between the fire and flooding disasters are summarized in the following table.

Model	**Training score**	**Validation score**	**Cross-validation score (10 folds)**
LinearSVC	0.983	0.981	0.982
RFC	0.996	0.981	0.986

Across almost all three metrics, the RFC model performed marginally better over the LinearSVC model. This is presumably due to the finely-tuned decision-making process of RFC models, being able to draw clearer distinctions between a Midwest flooding and SoCal fire. It is likely that, similar to the exploratory PCA, the LinearSVC model was unable to generate a clear hyperplane that defines the separation of data points. The normalized confusion matrices (Supplamental Materials pg. 13-14) describe the ratio of labeled data across each row, with true positives and negatives being maximized and false positives negatives being minimized.

5.2 Task B

Training, validation, cross-validation accuracy scores, and F1 score for classifying between the damage levels of Hurricane Matthew images are described in the following table.

Model	Training score	Validation score	Cross-validation score (10)	F1 score
LinearSVC	0.574	0.578	0.574	0.569
RFC	0.748	0.631	0.624	0.628

The weighted average of F1 scores, which are more insightful for multi-class datasets, are emphasized over accuracy for Task B. The rationale behind this is that adequate accuracy could be achieved if a model were to assign each image with the majority label. With the F1 score representing the harmonic mean between precision and recall, it is a more robust measure of predictive performance. The difference in accuracy metrics are much larger compared to Task A, with the RFC outperforming the LinearSVC model once again. These results seem to indicate that the RFC is much more suited for differentiating between disaster levels and multi-class tasks overall, considering that similar features across adjacent damage levels may make it harder to define clearer support vectors that can separate images. Additionally, the RFC outperforms the xView2 ResNet50 model's performance of 0.2654, albeit on vastly different input images and features.

6 Discussion

6.1 Model Appraisal

Task A proved to be a success, with the most constitutive adjustments to the model coming from the identification of ideal features and hyperparameters. Achieving accuracy metrics as high as 98% across both models and all splits indicates that both models are effective in distinguishing images with clearer, more differentiable color features.

On the other hand, Task B saw middling F1 scores and lower accuracy scores, as the task to identify damage labels proved challenging. Balancing the damage labels did provide more samples for training and improve the F1 scores. The most important features seemed to come from the image processing step, with LBP and Gabor features highlighting textural differences between the images and damage labels. Still, the current model is an improvement upon past models and could be further optimized to extract more impactful features that emphasize edge detection and subtle color differences.

Overall, the RFC model proves to be the best for both tasks, due to its ability to parse through features and draw distinctions between different classes.

6.2 Project Extensions

The relatively successful outcome of these models suggests that simple, high-performing computer vision models do not always require the use of costly CNNs. In a related article comparing the performances of CNNs and random forests for image segmentation, the loss in accuracy and weighted-mean F1 scores is made up for in the model building step, as "80 epochs for the U-Net CNN took approximately eight hours on a single GPU machine, while model builds for Random Forest on a single CPU took approximately five minutes" [Bos22].

The advantage of using a RFC model lies in assuming the middle-ground between performance and accuracy: being able to classify a high percentage of images correctly, whether it be binary or multi-class, while being not as costly in terms of machine resources and runtime. Although the runtime for training the RFC was longer than the LinearSVC, the increase was not magnitudes larger, as would be the case with a CNN. The improvement in performance can only be further optimized by collecting more relevant features and tuning other hyperparameters that this work was unable to explore. Future models and pipelines could also expand upon the feature-selection step, as well as applying PCA to potentially increase model accuracy.

References

[Tra98] Trapp. *Active Stereo Vision*. Nov. 1998. URL: `https://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/TRAPP1/filter.html` (visited on 04/22/2024).

[Fis+03] Fisher et al. *Feature Detectors - Sobel Edge Detector*. 2003. URL: `https://homepages.inf.ed.ac.uk/rbf/HIPR2/sobel.htm`.

[Pie10] Matti Pietikäinen. "Local binary patterns". In: *Scholarpedia journal* 5.3 (Jan. 1, 2010), p. 9775. DOI: `10.4249/scholarpedia.9775`. URL: `http://www.scholarpedia.org/article/Local_Binary_Patterns`.

[Bos22] Tony Bostom. *Comparing CNNs and Random Forests for Landsat Image Segmentation Trained on a Large Proxy Land Cover Dataset*. July 14, 2022. URL: `https://doi.org/10.3390/rs14143396`.

7 Supplemental Materials

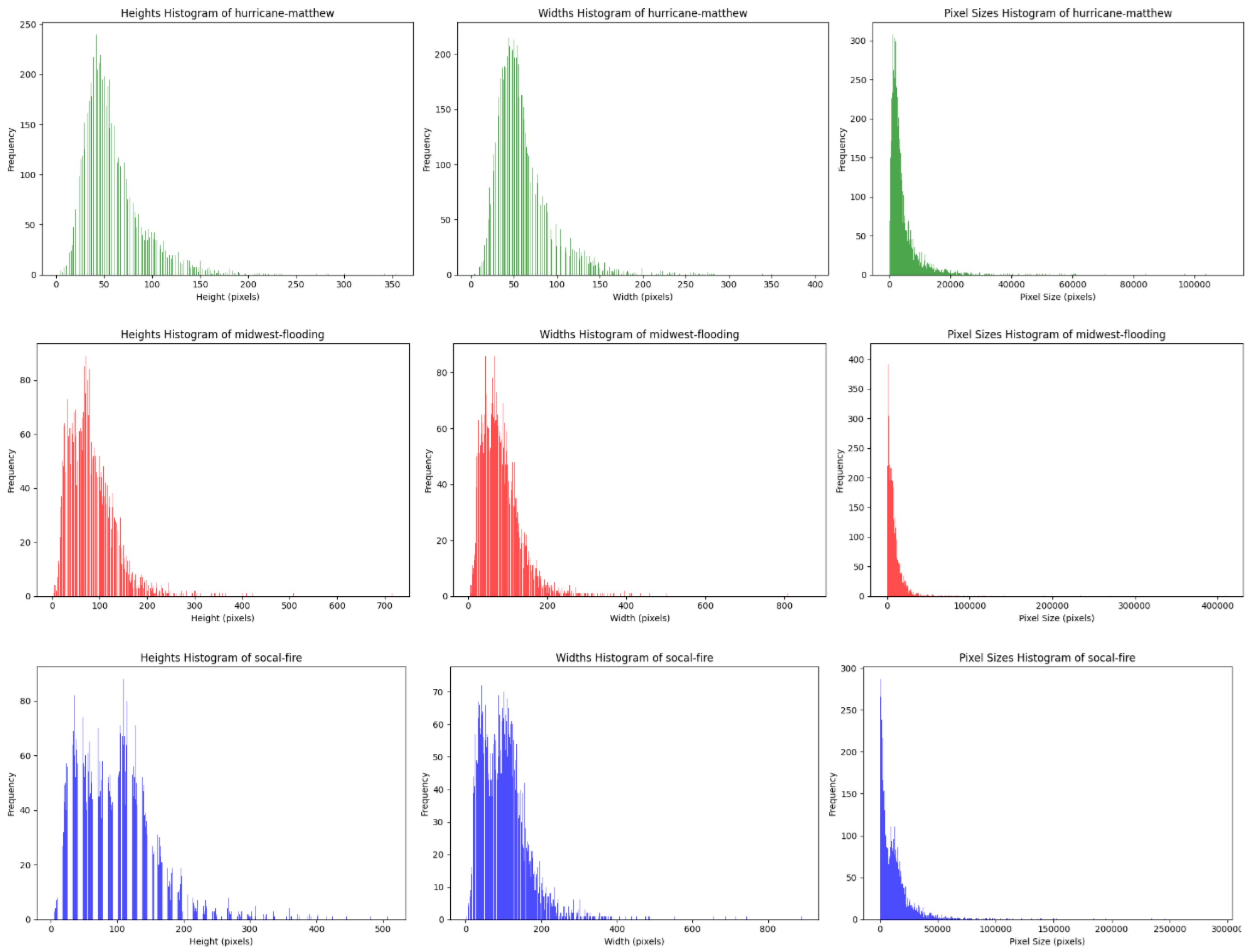

Figure 8: Distribution of height, width, and pixel size across 3 disaster types

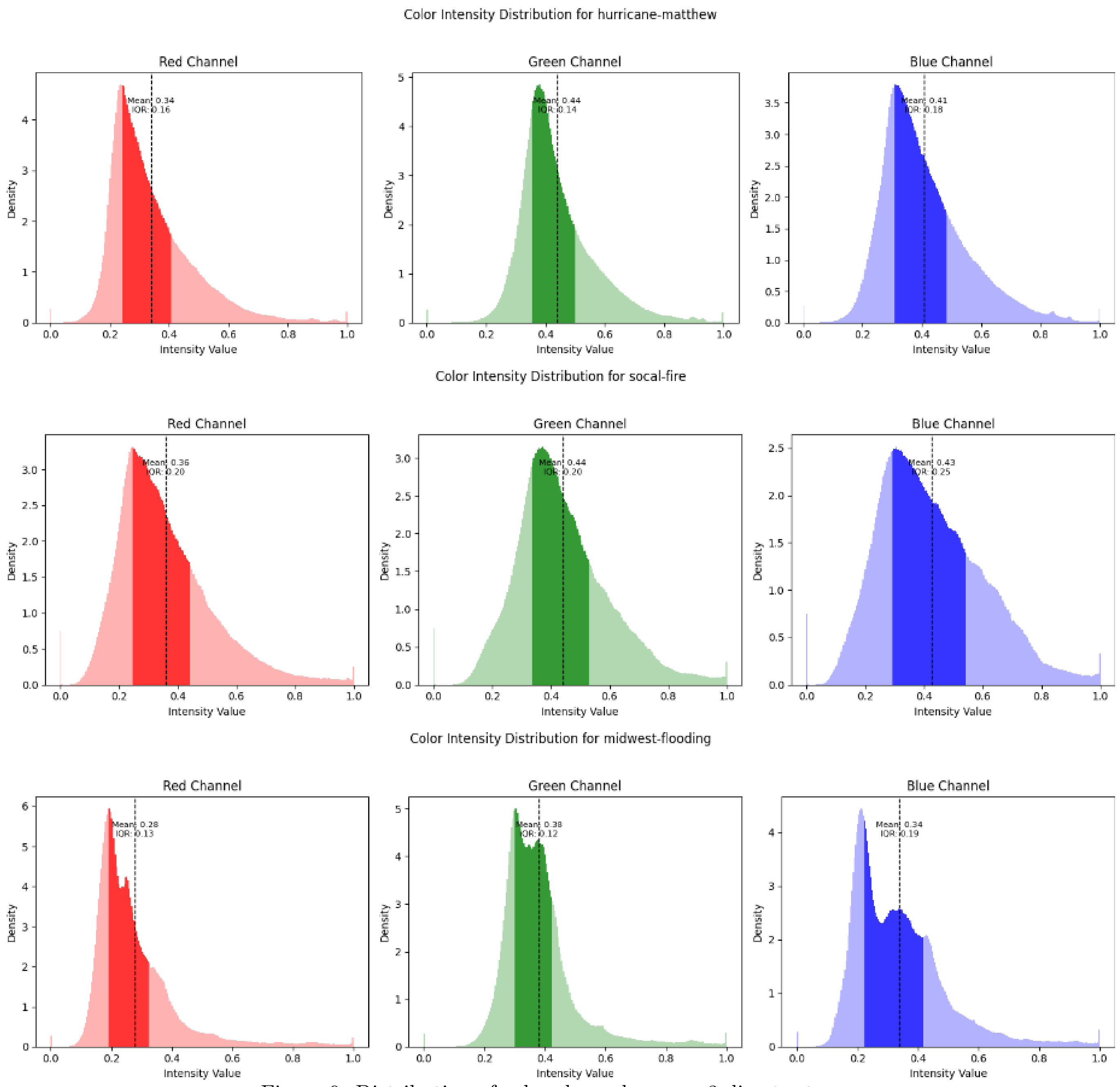

Figure 9: Distribution of color channels across 3 disaster types

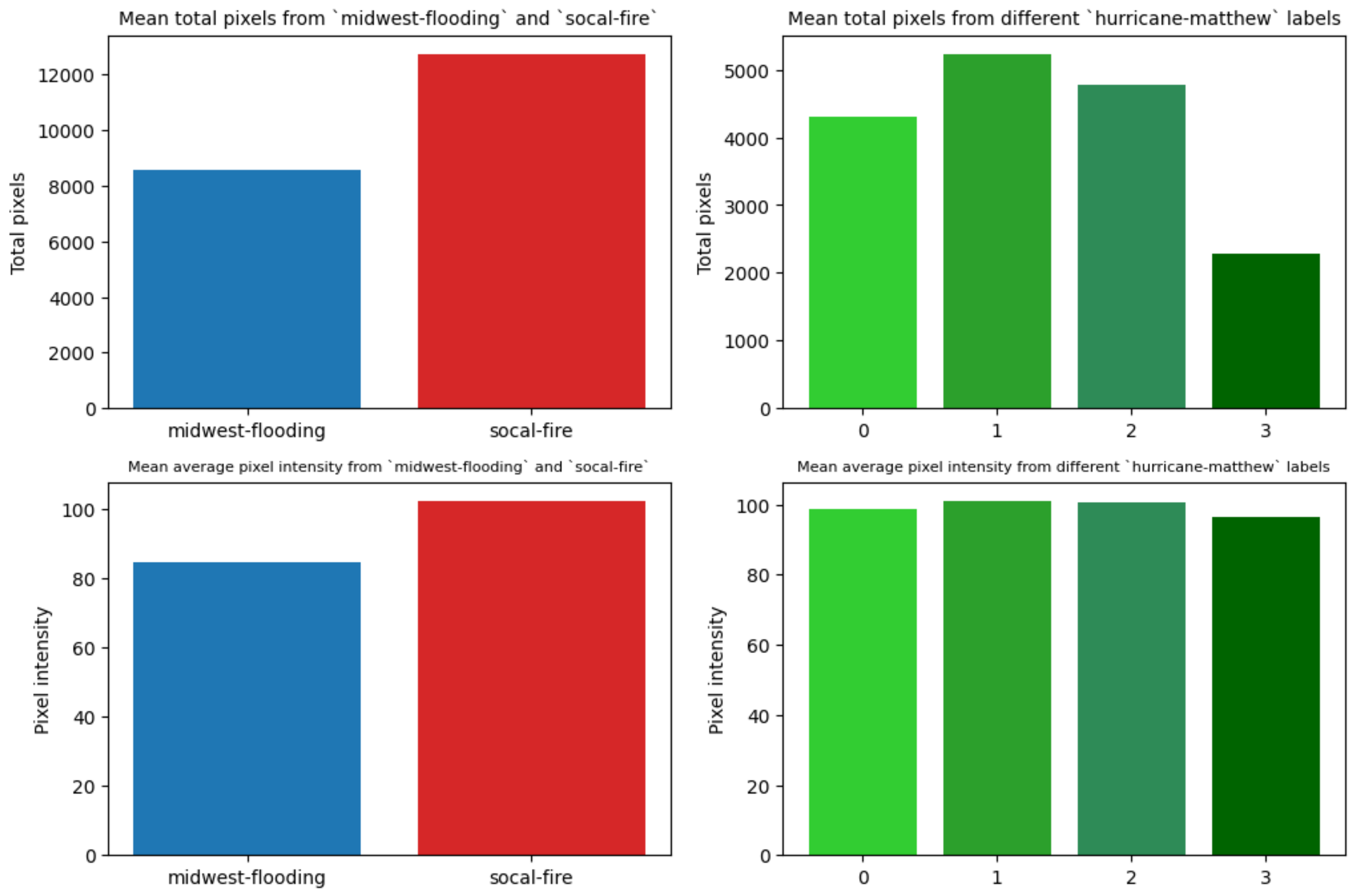

Figure 10: Results of GroupBy operations across three datasets

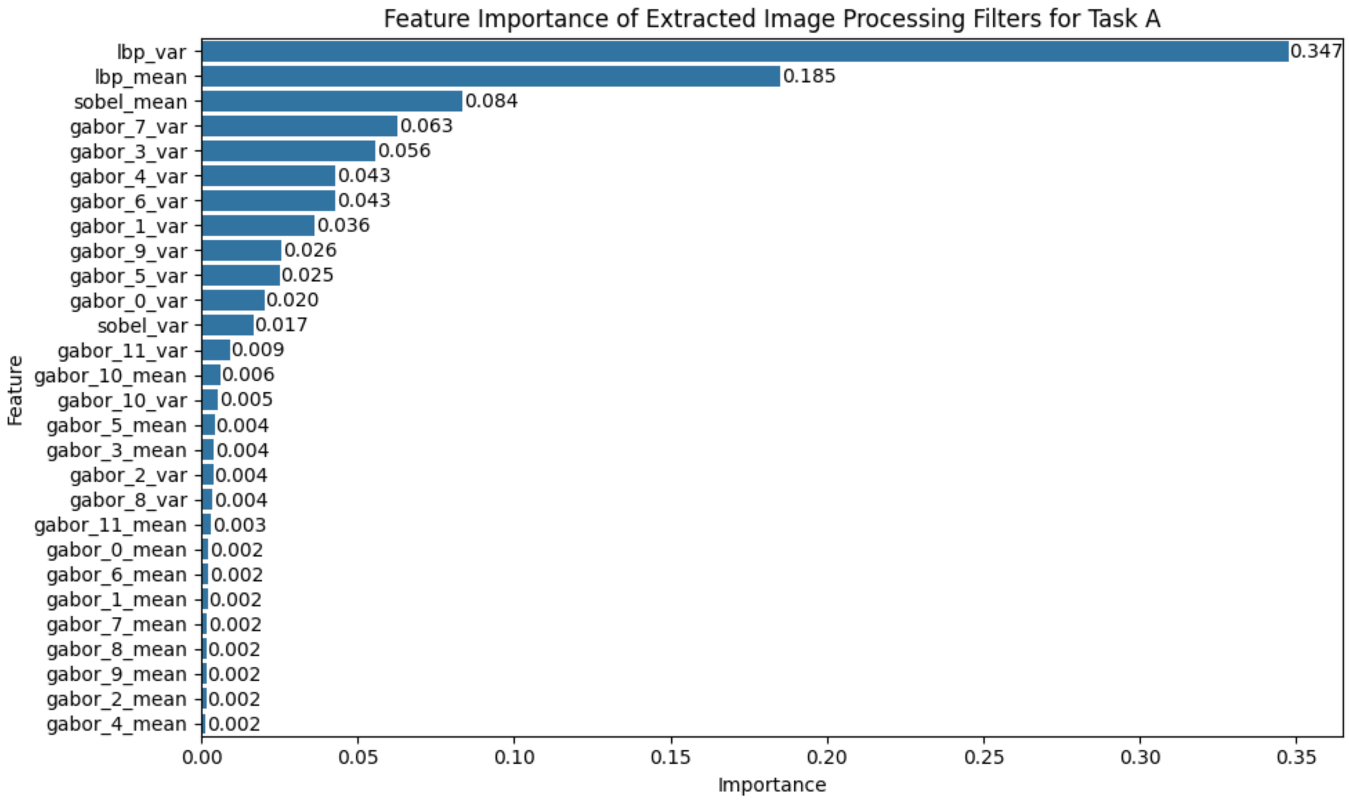

Figure 11: Feature importance for Task A

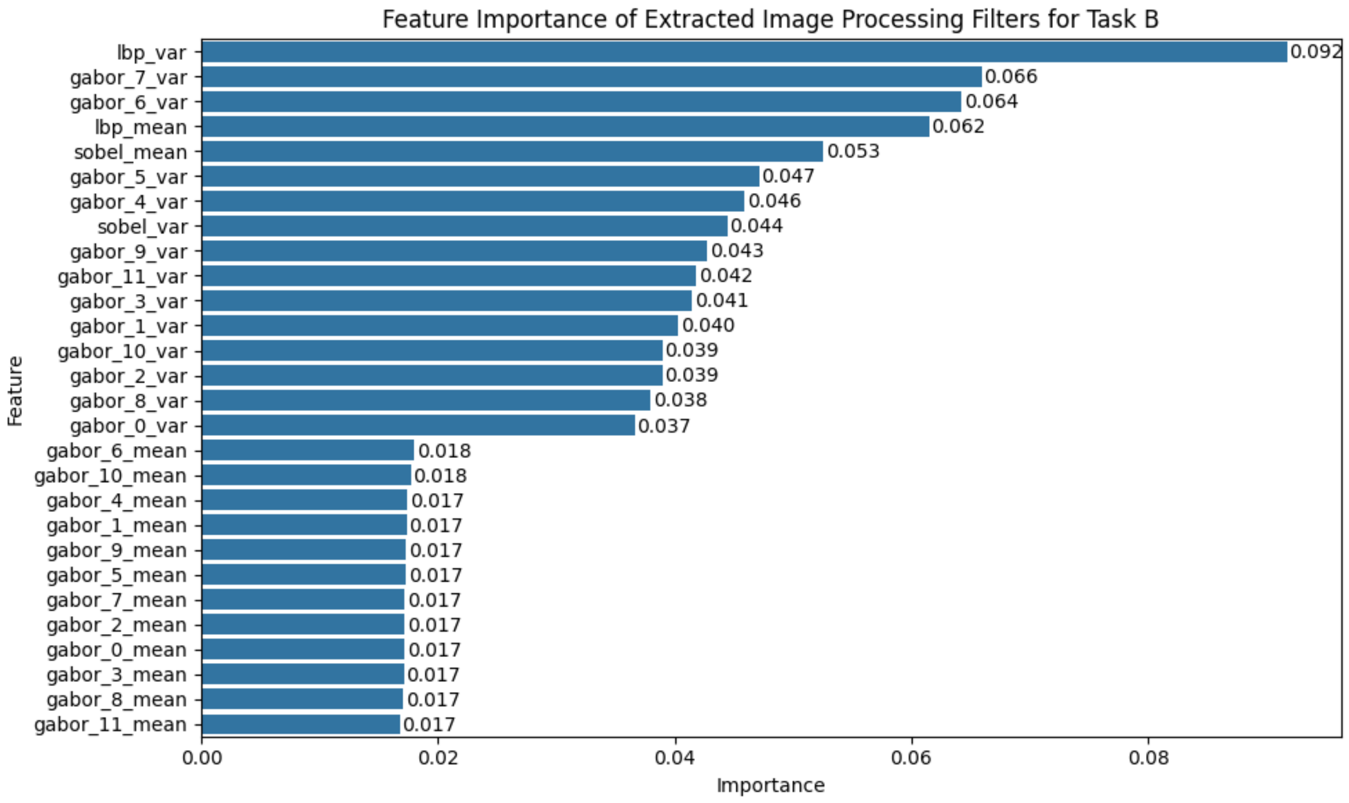

Figure 12: Feature importance for Task B

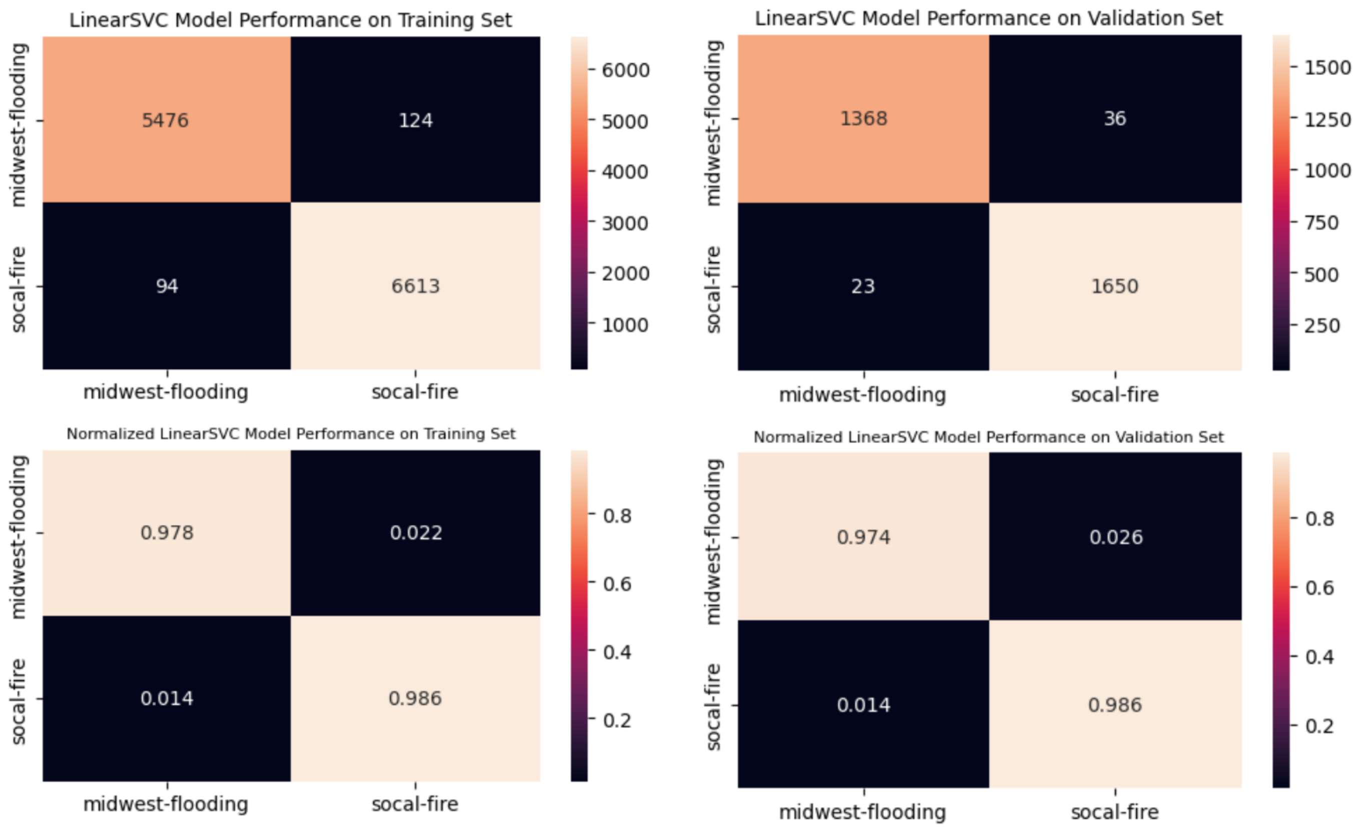

Figure 13: Confusion matrix for Task A, LinearSVC

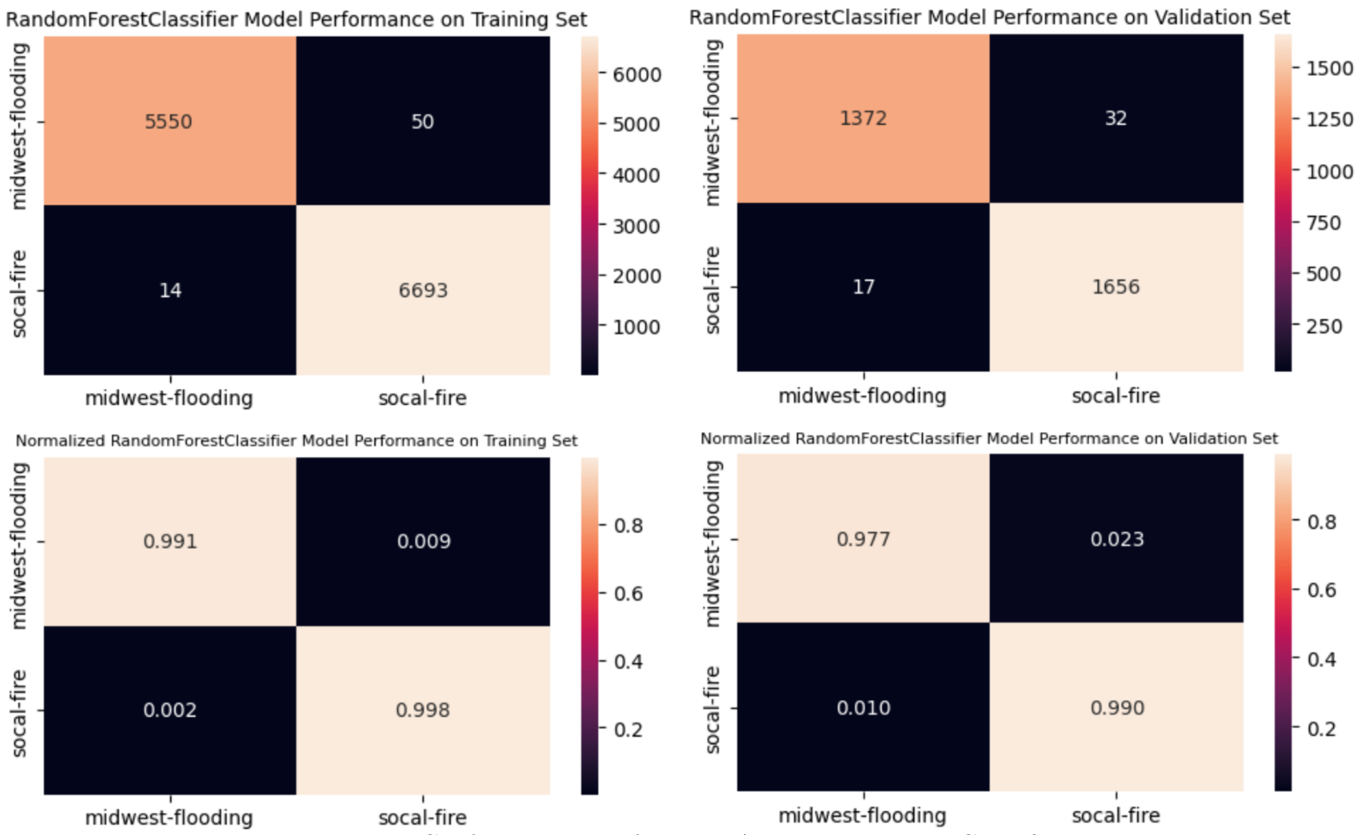

Figure 14: Confusion matrix for Task A, Random Forest Classifier

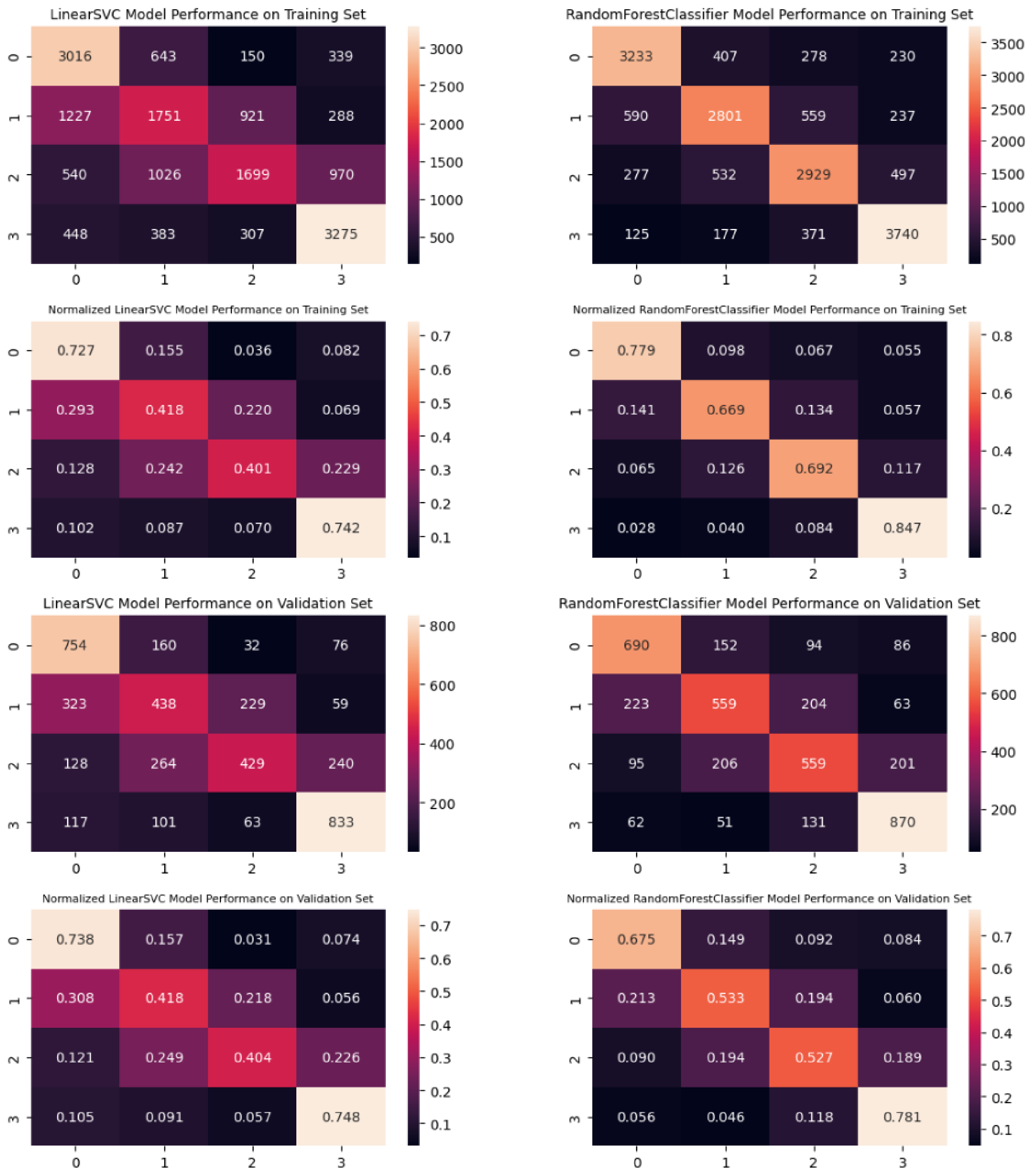

Figure 15: Confusion matrices for Task B, Random Forest and LinearSVC