UNIVERSITÀ
DEGLI STUDI
DI PADOVA

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

Final Project Report
Weather Classification with Machine Learning

STUDENT
Alexandru Mihailescu

ID NUMBER
2128831

1. *Introduction*

The project focuses on the weather image classification of MWD dataset. The scope of the project is to implement a solution from scratch which correctly identifies various weather conditions with reasonable precision.

Various approaches were considered to address this challenge, varying from traditional Computer Vision techniques to the application of advanced Deep Learning technologies. The chosen approach involves extracting meaningful features from the training dataset and inputting them into a Support Vector Machine (SVM) for subsequent data analysis, including classification and regression analysis. The trained model is then utilized to categorize unknown testing data into four distinct classes: "Cloudy," "Sunshine," "Rainy," and "Sunrise," as specified in the dataset.

An empirical approach to feature extraction was employed, beginning with a basic statistical representation of color space data in the form of histograms. The feature set was iteratively adjusted based on changes in the accuracy of the model on the testing data. Ultimately, the emphasis was placed on feature quality over quantity. However, experiments were also conducted using the entire set of available features to assess the SVM's performance on unseen data.

The code was written in Python within a Jupyter Notebook, and a copy of the SVM and Principal Component Analysis (PCA) were exported and shared within the project for quick implementation without the need for re-fitting or parameter optimization.

2. Dataset, libraries and declared code

The Multi-Class Images for Weather Classification (MWD) dataset utilized for training comprises a total of 1125 images. Within the 852 images of the training dataset, 7 could not be properly opened using the OpenCV *"imread"* function, resulting in uninitialized variables. Consequently, these images were excluded from the feature extraction and SVM fitting process. Similarly, 5 out of the 253 images in the testing dataset were also uninitialized, resulting in a total of 1113 valid images for this project's purposes.

From the wealth of python libraries, *Matplotlib* was used for generating inline visualizations of the test pictures, output examples and misclassification examples. *Scikit-learn* and *seaborn* libraries were used to acquire the Confusion Matrixes for the training and testing data, as well as corresponding heatmaps to visualize the performance of the SVM. The library *Scikit-Learn* was also necessary for the SVM, for the Principal Component Analysis (PCA) and for storing/loading of saved copies of both through *joblib*. Other libraries used were *OpenCV, Numpy* and built-in python libraries like *os, time* and *random* which were used for extracting picture paths, calculating performance of feature extractions, and creating some random output examples.

External code used that needs declaring corresponds to the "*show_img*" function present in a previous Computer Vision laboratory, the "*calculate_HSV_histograms*" and "*calculate_BGR_histograms*" which were inspired from the official OpenCV tutorials, as well as the "*get_averageAndDominantColors*" and "*display_averageAndDominantColors*" which were adapted and optimized for performance from a detailed *Stackoverflow* post on Average and Dominant colors. Precise details on source of the declared functions are found in the Jupyter notebook.

3. Methodology

As mentioned in the introduction, the approach to feature selection was primarily guided by empirical analysis of score values on unseen data. *Table 1* presents not only the score values obtained for the trained SVM on testing data, but also the gradual process of inclusion and exclusion of features, based on their impact on the final score.

The testing of the SVM commenced with the utilization of HSV histograms. These were initially assessed as containing more useful information than the BGR histograms for the SVM. While the initial score was not very promising, the next

step was that of extracting as features the image contrast, the average color of the picture, and six dominant colors. Dissatisfied with the performance, the decision shifted to using the BGR color space histograms and found that contrary to initial assumptions the SVM was performing better than with the HSV color space histograms.

BGR histograms			✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
HSV histograms	✓	✓													✓	✓
Contrast		✓	✓	✓	✓	✓		✓	✓			✓	✓	✓	✓	✓
Average Color		✓	✓	✓	✓	✓			✓	✓		✓			✓	✓
Dominant Colors		✓	✓	✓	✓	✓				✓	✓	✓	✓	✓	✓	✓
DoG BGR histogram					✓										✓	✓
DoG HSV histogram						✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
DoG Average														✓	✓	✓
DoG Contrast														✓	✓	✓
PCA trasformation				✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓
Testing Score	78.42%	80.94%	82.22%	84.29%	93.53%	95.32%	93.88%	93.88%	93.88%	94.96%	94.96%	95.32%	**96.04%**	94.96%	94.24%	94.60%

Table 1 - Features used for training and scores on Testing dataset.

At this stage, the number of features fed to the SVM was 797 for each picture and meant only to increase with the addition of other features. To reduce the dimensionality of the data, and hopefully make it easier for the model to generalize from the data, a PCA was implemented with a fixed number of features of 200 (a value that was also determined through experimentation and was found to be small enough that it does not degrade the performance of the SVM). A small increase in the testing score can be seen in response to the PCA transformation of the features in *Table* 1.

To further enhance SVM performance, the addition of the Difference of Gaussian (DoG) feature was considered, as it's an enhancement algorithm capable of providing valuable information for better classifying images. Instead of applying DoG to a grayscale version of the image, it was opted to apply it to the BGR color space and subsequently extract and visualize the histograms while monitoring SVM performance. Since the image classification problem in this case can be reduced to a color analysis problem, eliminating the color information from the DoG seemed like it would take away potentially useful information that is not immediately obvious to a human, that a SVM could find useful. In fact, the addition of the DoG BGR histogram led to an approximate 9% increase in the testing score, indicating the capacity of SVM and PCA to extract meaningful patterns from it that significantly enhance image classification.

Upon comparing the DoG HSV and DoG BGR histograms, it was evident that the BGR Histogram contained redundant information consistent across all three channels and coinciding with the lightness channel of the HSV histogram, as depicted in *Figure 1*. This latest histogram also seemed to contain additional information from the DoG image that the BGR histogram did not contain and could benefit the classification process of the SVM. In fact, by switching from DoG BGR histograms to DoG HSV histograms in the picture features set, a further 2% increase in the SVM accuracy was observed.

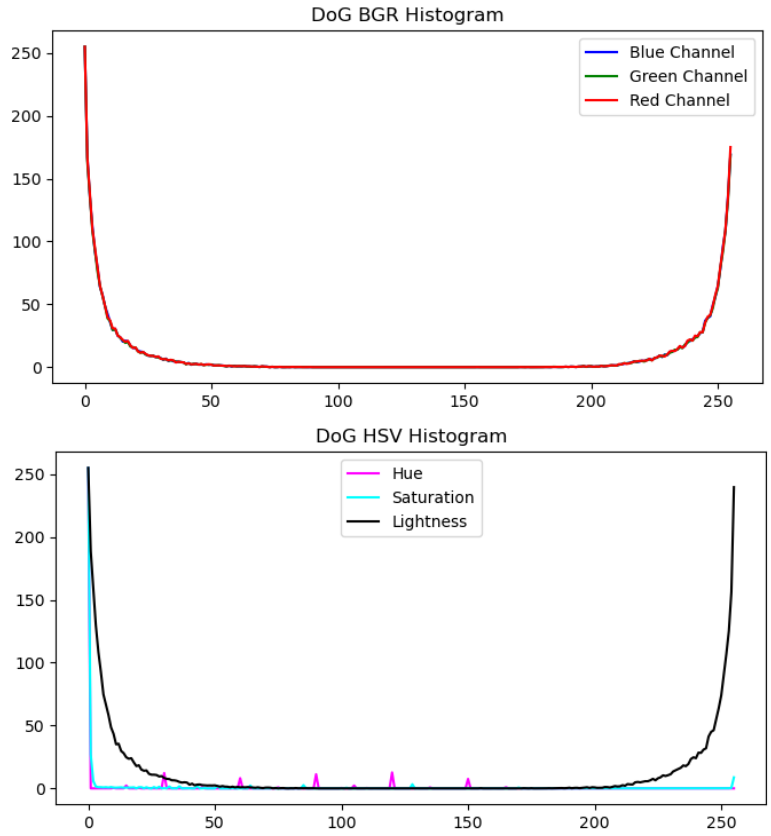

Figure 1 - Comparing HSV and BGR histograms of the DoG of a picture from the training dataset

At this point, the accuracy was already reasonably high and more than satisfactory. To assess the impact of various features on the accuracy values, gradual feature removal and re-introduction was conducted, as detailed in *Table 1*. Observations indicated that Contrast, Average Color, and Dominant Colors collectively increased the score performance by almost 2%. However, the maximum attainable score was achieved by removing Average Color information while retaining Contrast and Dominant Colors. This suggests that Average Color potentially introduced redundant information that was not easily eliminated by PCA, possibly complicating the final model's ability to distinguish between classes.

As part of additional tests, DoG average color and contrast information was included to the features that had already demonstrated high quality in SVM training. This addition resulted in a slight decrease in performance, similar to the impact observed when including Average Color information earlier. Finally, the SVM was tested with all extractable features, both with and without PCA transformation, resulting in a slight decrease in scores, as was anticipated from the addition of so many features.

4. Notebook summary and brief explanation

The Jupyter notebook was divided into four major sections for clarity, which were also divided into subsections based on their function. The notebook's ability to collapse and expand whole sections allows for improved readability and ease of navigation in the code, which made it easier to debug and implement functions. The general structure is as it follows:

1. ***Module import and basic functions***
   - 1.1. Libraries and modules necessary in the notebook
   - 1.2. Global variables such as the absolute dataset path
   - 1.3. A binary variable that can be changed to allow loading of the pre-stored SVC and PCA as a fallback.
   - 1.4. The *"show_img"* function, which allows for inline viewing of a picture, and the *"get_imagePaths_byWeather"*, which returns a list of absolute paths for pictures of the training or testing datasets, based on the specific desired weather condition.
2. ***Functions used for feature extraction***
   - 2.1. Contains a simple Z-normalization function.
   - 2.2. Functions for BGR histogram calculation and visualization (that can also be used on a single test picture)
   - 2.3. Functions for HSV histogram calculation and visualization
   - 2.4. Implementation of adapted code that returns picture average color and the 6 most dominant colors.
   - 2.5. Picture contrast calculation
   - 2.6. DoG picture calculation, through the difference of two versions of the picture where Gaussian Blur is applied with different strengths.
   - 2.7. Finaly, a *"get_picture_color_features"* function which combines all the above to output a single one-dimensional NumPy array.
   - 2.8. This last subsection extracts 4 random pictures from the testing database, one for each weather condition, and applies to it the previous functions, while also visualizing the results of these calculations and

resulting feature size. This section was also useful for debugging the functions and checking if they behaved properly.

3. Creation of Training and Testing dataset with PCA reduction

3.1. The dataset creation function "*create_inputOutput_dataset"* is defined in this section, which calls the 2.7 "*get_picture_color_features"* function for the specified dataset type (training or testing) and desired weather conditions (while also allowing create a dataset with reduced weather classes if wanted).

3.2. After getting the number of features from a test picture, this is fed to the "*create_inputOutput_dataset"* to generate both training and testing datasets.

3.3. The input X datasets are then transformed with a 200 features PCA, or, in the case in which the 1.3 "*use_pre_storedSVCPCA"* variable has been set to 1, allows to load the pre-saved PCA, which was fitted on the optimal features from *Table 1.*

4. SVM training and model accuracy assessment

4.1. The same goes for the SVM. After checking for the value of the 1.3 "*use_pre_storedSVCPCA"* variable, the block will either load the saved model or start a GridSearch for the optimal SVC parameters by employing all available CPU processors and displaying the best parameters and score results.

4.2. The SVM is then used to create confusion matrixes and heatmaps, which are plotted together with the score results for training and testing datasets.

4.3. And then it is used to generate a random from output example from the testing dataset for each weather condition, together with the ground truth and predicted class.

4.4. Finally, an example of misclassified outputs is given and shown, for confrontation.

5. Model evaluation and results

As mentioned earlier, the main method of assessment of the SVM performance was done through Score calculation and Confusion Matrix heatmap confrontation.

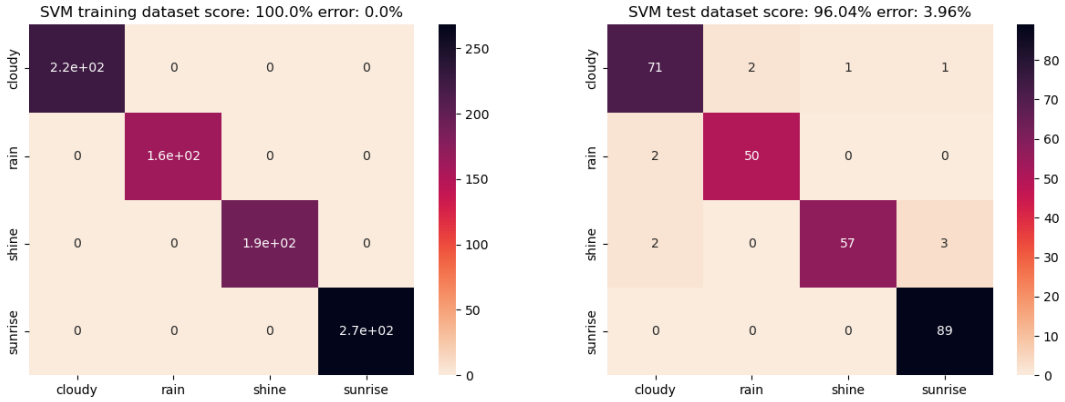

Figure 2 - Best score SVM heatmaps on MWD dataset

For the highest quality features found and displayed in *Table 1* derived a SVM with a training dataset score of 100% and a very satisfactory test dataset score of 96.04%, as shown in *Figure 2*. The higher training score could be a sign of slight overfitting, but considering the that the difference is of just 4%, it cannot be conclusively determined just from the score difference.

The model performed well on unseen data, with the main confusion done between cloudy and rainy pictures, but also between "shine" and "cloudy", as well as "shine and "sunrise". "Sunrise" pictures from the testing data, on the other hand, were never misclassified. Some output example pictures of correct and incorrect classifications can be found in the next page, and further examples (for the ACDC dataset as well) can be found in the "Output examples" folder delivered with this report.

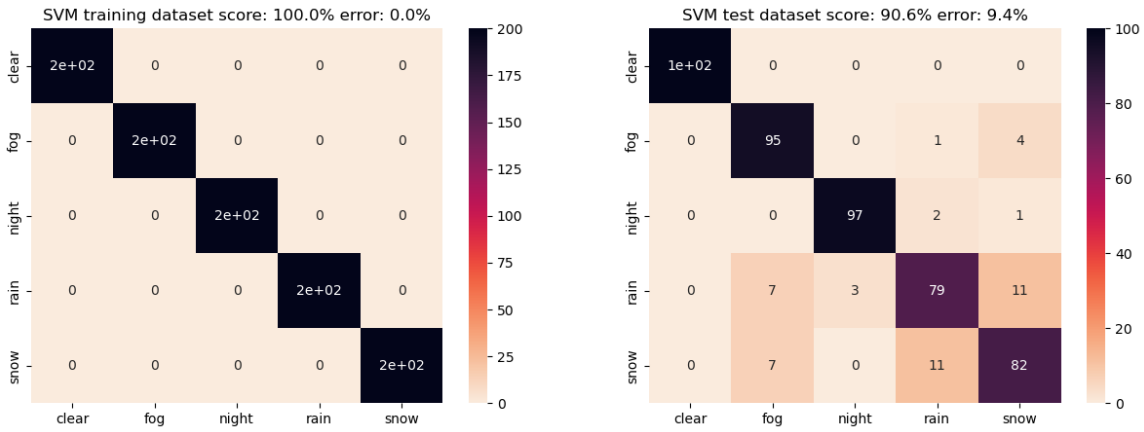

Figure 3 - SVM heatmaps on ACDC dataset

Out of academic interest, the code was also adapted and tested on the Adverse Conditions Dataset with Correspondences (ACDC) dataset with the above-mentioned features. The performance of the fitted SVM is visible in *Figure 3*.

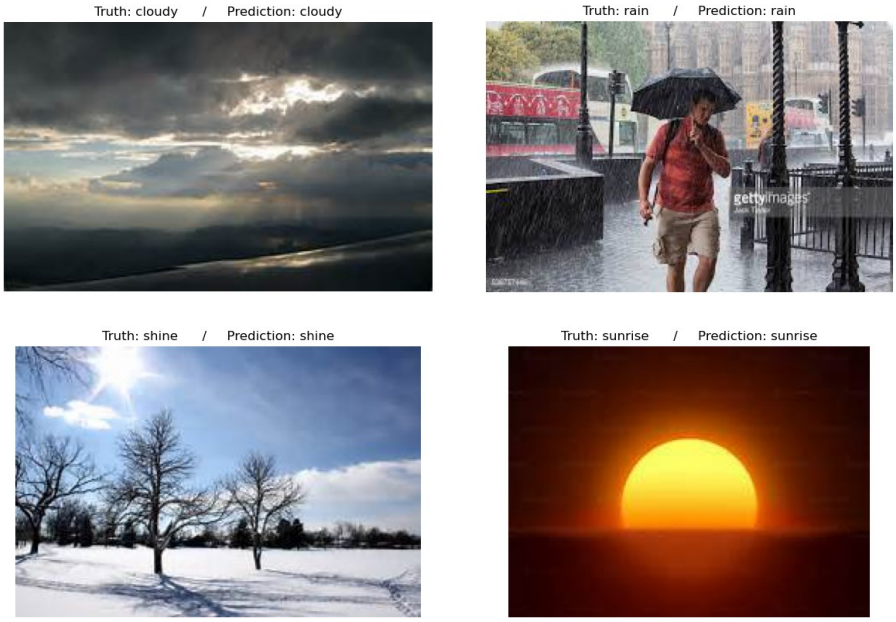

Figure 4 - Examples of correct classification outputs on the MWD dataset

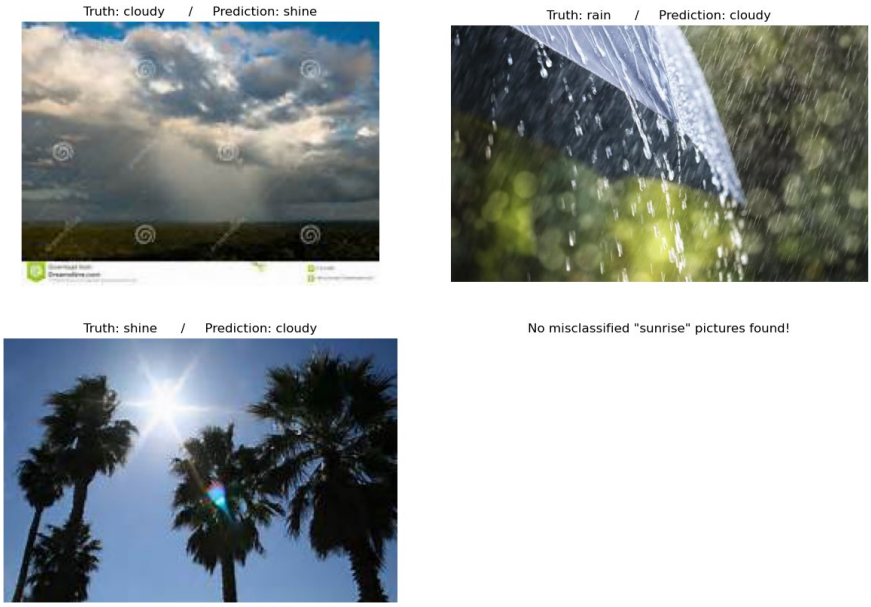

Figure 5 - misclassification examples on the MWD dataset

The model performed reasonably well on unseen data, even though the extracted features were not being tailored for this specific dataset. The overfitting of the SVM is also more noticeable with this dataset than with the MWD dataset, seen in a training score of 100% and a testing score of 90.6%. Confusion between the

classes “rain” and “snow” / “snow and rain” are the more prevalent with the model, followed by misclassification of the classes “rain” and “fog”. This outcome was anticipated since the extracted features were primarily designed for color information analysis in datasets with distinct color differences between classes.

In summary, the SVM demonstrated impressive performance on the MWD dataset, achieving a high level of accuracy on unseen data. While it was also tested on the ACDC dataset, its performance was slightly compromised due to the feature set's specialization in color-based analysis.

6. Conclusion

Some further possible developments of this project are spotted in the difference between datasets. Looking at the misclassification example in *“Output examples/ACDC_misclassification_example.png”*, it becomes evident that the ACDC dataset poses challenges due to less distinct color differences between classes compared to the MWD dataset. This similarity makes it more challenging for the model to accurately distinguish between very similar images.

A possible improvement to the model, that is also a more complex approach to the problem, could involve implementing sematic segmentation of the ground and subsequent color analysis. By analyzing pictures from the ACDC dataset, this approach would seem relevant since the sky in both “snow” and “rain” pictures are very similar in color contents, but the ground in “snow” has an abundance of white in its dominant colors. In addition, conducting semantic segmentation of the sky and analyzing its color could help the model perform accurate multiclass image classification between not only different weather conditions but also different periods within the same day.

In conclusion, this project has successfully achieved its intended scope, yielded highly satisfactory results while also opening the doors to interesting opportunities for further development and research.