Optical character recognition EDA

Aryan Khatana (aryan21237@iiitd.ac.in)

Parthiv Dholaria (parthiv21078@iiitd.ac.in)

Pankaj Ramani (ramani20234@iiitd.ac.in)

Shubham Kale (shubham23094@iiitd.ac.in)

Pragati Agrawal (pragati23059@iiitd.ac.in)

Abstract

This report presents a comprehensive Optical Character Recognition (OCR) project employing a variety of machine learning techniques. The EDA process comprised of grayscale conversion, image resizing, class imbalance check, mean and variance calculation, plotting pixel intensity distribution curves, zernike moments calculation, edge detection, generating local binary pattern (LBP) Images, histogram of oriented gradients (HOG) Images, mean pixel intensity line graphs, principal component analysis (PCA), Explained Variance Analysis. The methodology encompasses data preprocessing, feature extraction, and modeling using traditional machine learning algorithms. Augmentation techniques, including flipping, rotation, noise addition, and zooming, enhance the dataset. Feature engineering involves extracting pixel values, Histogram of Oriented Gradients (HOG), and Local Binary Pattern (LBP) features. The models include a custom multiclass logistic regression, Random Forest, Support Vector Machine (SVM), and K-Nearest Neighbors (KNN). Stacking is applied to combine the strengths of individual models, achieving an accuracy of approximately 99.6 percent on the test set. The report discusses the analysis and results.

1 Methodology

Following are the steps that were followed:

- Exploratory data analysis and data pre-processing.
- Freature extraction, training 4 models namely Support Vector Machine (SVM), Multiclass Logistic Regression, Random Forest, K-Nearest Neighbors (KNN) which gave us decent accuracy.
- Further, k-fold validation, feature engineering and boosting improves the accuracy.
- Moreover, stacking approach combines the strengths of multiple models, potentially improving overall prediction accuracy.
- Furthermore, PCA, data augmentation are performed making our model more robust.

Preprint. Under review.

1.1 EDA and Data Pre-processing

1.1.1 Class Imbalance Check

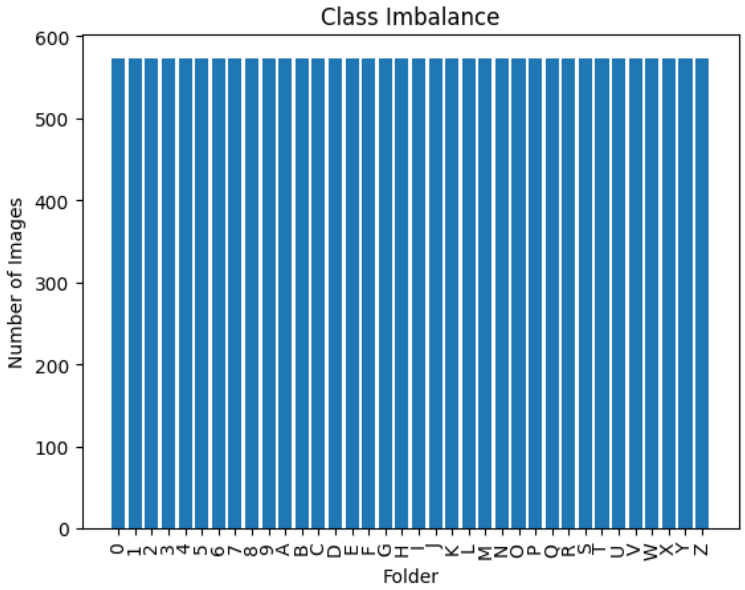


Figure 1: Number of Images vs Folder(Class).

From the graph it is clear that there is no class imabalance as each folder contains the same number of images for the train data so there is no need of performing oversampling or undersampling.

1.1.2 Mean, Variance and Standard Deviation

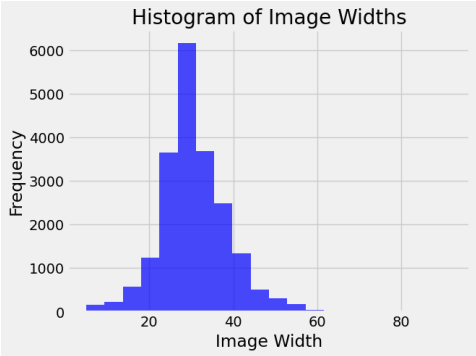
We have a dictionary that contains mean, variance and standard deviation for each class which can be found at `https://drive.google.com/file/d/1029H_Ey4mLQnpdVgVRory0Lhe35F0je4/view`.

The mean of all the classes present in this dataset are similar and so from this we can infer that:

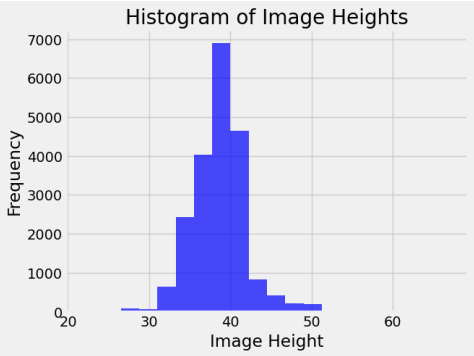
- The dataset is well-balanced
- These classes have similar statistical characteristics or properties. This means that the classes represent entities with similar underlying features.

Similar standard deviations and variances suggest that the data points within different classes exhibit comparable levels of variability or dispersion around their respective means.

1.1.3 Resizing



(a) Figure 2(a)



(b) Figure 2(b)

Figure 2(a) represents frequency vs width histogram and it peaks at 28. Figure 2(b) represents frequency vs height histogram and it peaks at 38. Therefore, the best size to reshape all the images to is (38.0, 38.0).

1.1.4 Pixel Intensity Distribution Curves

The pixel intensity distribution curves can be found at `https://drive.google.com/file/d/1DXjiOJ99zfzk-p3YhiGGCXZnM7NwtEuK/view`. It can be seen from the above graphs that all the images in a class have similar pixel intensity curves from which we can infer the images in the class have a consistent distribution of pixel intensities.

1.1.5 Edge Detection

The Edge Detection plots can be found at `https://drive.google.com/file/d/1B4n7XhQapIohK5JeW4DhOseMmODfQP7s/view`. By looking at the images of each class after performing the edge detection we can infer that there are distinctive edge patterns among the classes. Each character and digit presents a unique edge pattern. The sharpness, continuity, and layout of these edges are distinctive for each class. For instance, the number "8" has two clear circular patterns, while the letter "A" has a distinct peak with a horizontal crossbar in the middle i.e. "A" has a continuous edge patterns where in case of a digit number 8 has not a straight edge pattern but instead a circular edge pattern. This can help us in differentiate between class "8" and class "A" similarly it can be extended to other classes as well. Hence it is very crucial to study the Edge patterns in differnet classes.

1.1.6 Local Binary Pattern (LBP)

The Local Binary Pattern (LBP) plots can be found at `https://drive.google.com/file/d/1muGyfXefedN86ywyHh8cCmeSJ97rf5wA/view`. LBP plots provides a compact representation of the image, where the local patterns can be captured with just a few bits per pixel. This reduces the dimensionality of the data, making it quicker and often more accurate for classification tasks.

1.1.7 Histogram of Oriented Gradients (HOG)

The Histogram of Oriented Gradients (HOG) images can be found at `https://drive.google.com/file/d/1hryRunuOto5KwNyxXrZzyNgdSlhygkyg/view`. HOG analysis helps us in analysing the noise in the images. HOG captures the intensity of gradients. Areas with very faint gradients that don't match the general structure of the digit could be a result of noise. Very light or inconsistently dark patches might be indications of areas with less meaningful information or noise. Consider a scenario to classify class 7 and class 8. There is sufficient amount of noise in class 7 that can help me classify between class 7 and class 8

1.1.8 Noise Addition and Data Augmentation

Random noise is added to images to improve model robustness. Image augmentation techniques such as flipping, rotation, and zoom are applied to increase the diversity of the training data.

1.1.9 Mean Pixel Intensity

The Mean Pixel Intensity Line Graphs can be found at `https://drive.google.com/file/d/1oezkINkJumgaA8NyAyAXU8JpDxIaFfc4/view`. Here we can see individual class wise mean pixel intensity. When it comes to classification of characeter/digits Pixel Intensity graphs provide insights into the distribution of pixel values (intensities) in an image.

It helps us in classification in the following ways:

- The positions and heights of peaks in the histogram can provide information about the dominant pixel values, which might be characteristic of certain classes of objects or patterns.
- For example, Consider the case when you try to classify 8 and 9. Now look at the peak intensities graph of their respective classes.
- If we closely observe certain pixel positions, we can notice differences between the two graphs, around pixel positions 200-800: Group 9 seems to have a generally higher intensity fluctuation than Group 8. After position 800: Group 8 has higher pixel intensity fluctuations than group 9.
- Hence by analysing peak intensity fluctuation we can classify between classes easily.

1.1.10 Zernike Moments

Zernike moments are useful for characterizing and quantifying the spatial frequency components and geometrical features like average intensity or brightness, asymmetry, skewness, shape characteristics, rotational asymmetry, etc of an image. We have the Zernike moments of degree 5 and a dictionary that contains the zernike moments of 3 random images from each class which can be found at `https://drive.google.com/file/d/1exzhhHM1S-k1rN9oy-zhdSROJnkhCfwV/view`. From the zernike moments we can see that the elements of similar looking shape has some similar values that represents the shape characteristics of those elements. Other such properties can be verified by closely observing the dataset and mapping the values.

1.1.11 Principal Component Analysis (PCA)

Components Analysis

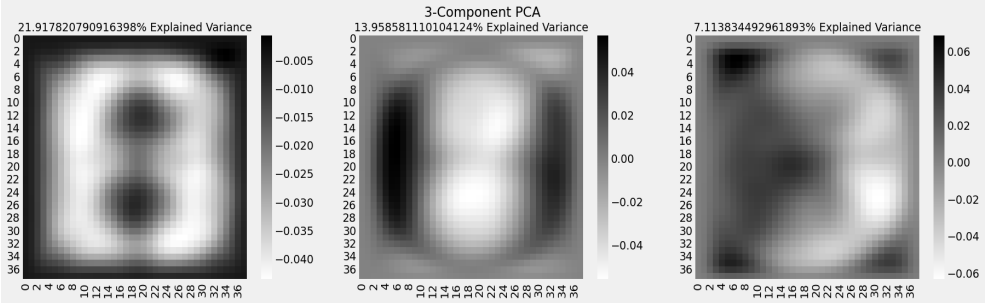


Figure 2: Component Analysis.

From the above output we can verify that the first component explains about 22 percent of the variance in the dataset, we can prety clearly see a shape that is common in classes like - B, H, 8 and so on. The 2nd component that explains about 14 percent of the variance shows curves like in 0 and O and some other clases as well. Each subsequent component explains less and less of the variation in the data.

Variance Analysis

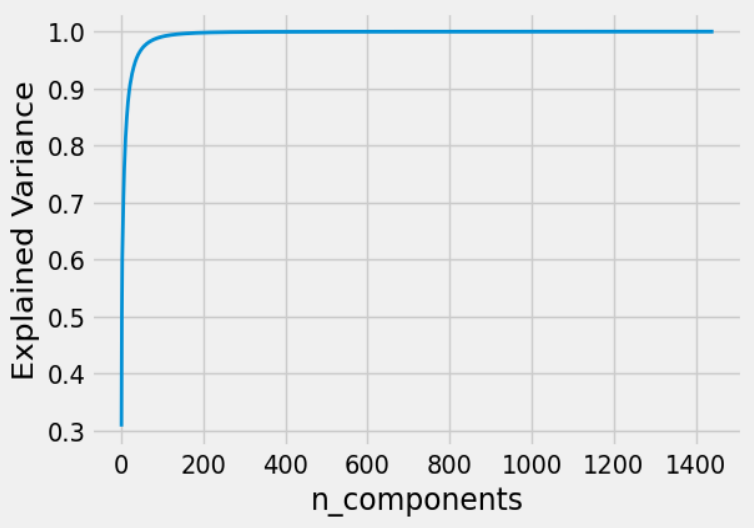


Figure 3: Explained Variance vs Number of Components.

We can see from the above graph that 100 percent of the variance of the data can be explained by just 200 components, this suggests that many of the original 1444 features may be linear combinations of each other. In other words, we have a high degree of dimensionality in our dataset, but much of the information can be effectively captured in a lower-dimensional space, exponentially decreasing the computational overhead while training. So we can reduce our data to 200 dimensional wihtout significant loss of information.

1.1.12 t-distributed Stochastic Neighbor Embedding

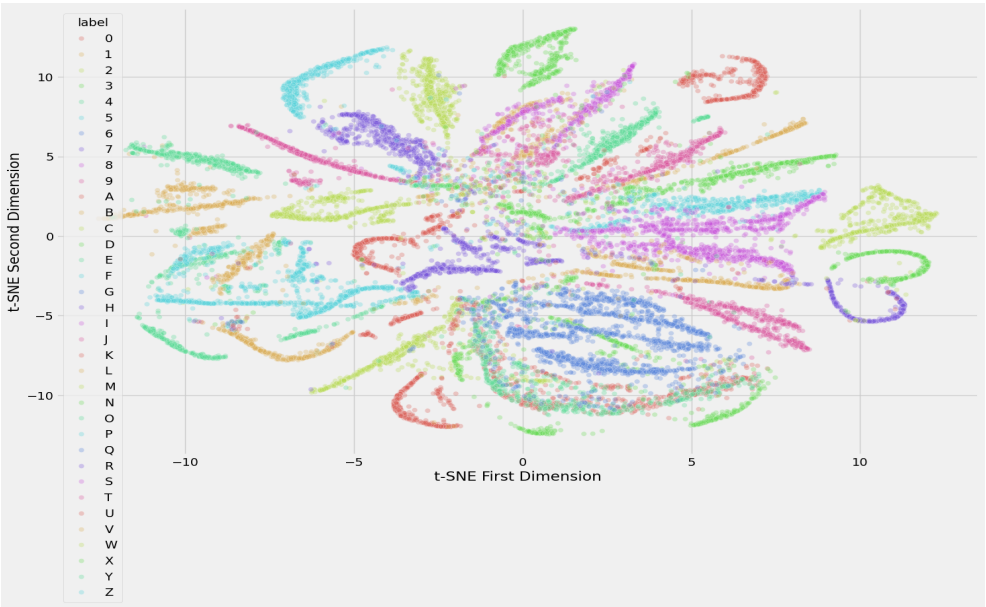


Figure 4: Explained Variance vs Number of Components.

t-SNE (t-distributed Stochastic Neighbor Embedding) is an unsupervised non-linear dimensionality reduction technique for data exploration and visualizing high-dimensional data. From the above graph we can see that each category is clearly clustered in its own distinct region.

From the EDA that we performed, features of LBP, HOG are used in the further model traning.

1.2 Model Development

The project employs a variety of models to address the character recognition task

1.2.1 Custom Multi-Class Logistic Regression Model

A custom logistic regression model is implemented to perform multi-class classification. The model is trained using stochastic gradient descent with a categorical cross-entropy loss function.

1.3 Random Forest

An ensemble of decision trees is trained to create a Random Forest classifier. The model is configured with 100 trees, a maximum depth of 10, and other hyperparameters for optimal performance.

1.4 Support Vector Machine (SVM)

A linear SVM classifier is utilized, configured for multi-class classification. The model is trained with probability outputs for later use in stacking.

1.5 K-Nearest Neighbors (KNN)

A KNN classifier with two neighbors is trained on the extracted features.

1.6 Stacking

Predictions from individual models are used as features to train a meta-model (SVM). This stacking approach combines the strengths of multiple models, potentially improving overall prediction accuracy.

2 Analysis and Results

- Multi-Class Logistic Regression: The custom logistic regression model achieves 98.90 percent accuracy.
- Random Forest: Achieves a commendable accuracy of 97.12 percent on the testing data.
- SVM: Demonstrates outstanding accuracy of 99.70 percent on the testing data.
- KNN: Achieves a good accuracy of 97.52 percent on the testing data.
- Stacking Approach: The stacking approach further improves accuracy, achieving 99.60 percent on the testing data.
- Final Accuracy: The final ensemble model, combining the strengths of individual models through stacking, achieves an impressive accuracy of 99.60 percent on the testing data.

3 Conclusion

The OCR project successfully applies a comprehensive methodology, encompassing data preprocessing, feature extraction, and model development, to achieve highly accurate character recognition. The stacking approach proves effective in enhancing overall performance by leveraging the strengths of individual models. The achieved accuracy suggests the practical applicability of the system in real-world scenarios.

4 Existing Analysis in OCR

In a seminal ranking by LeCun et al. [2] encompassing references up to 2012, various machine learning techniques were assessed for Optical Character Recognition (OCR). The evaluated approaches included:

- Linear Classifiers: Error rates ranged from 7.6% to 12%.

- K-Nearest Neighbors (K-NN): Achieving error rates between 1.1% and 5%.
- Non-Linear Classifiers: With an average error rate of about 3.5%.
- Support Vector Machines (SVM): Demonstrating error rates from 0.8% to 1.4%.
- Neural Networks (NN): Ranging from 1.6% to 4.7%.
- Convolutional Neural Networks (CNN): Showing promising results with error rates between 0.7% and 1.7%.

Notably, the incorporation of data augmentation proved beneficial. In the case of CNNs, the best error rate achieved was an impressive 0.95% with no distortions or preprocessing [4].

Despite the ascendancy of CNNs in this ranking, classical machine learning techniques still held their ground, achieving competitive error rates (under 1.0%). For instance:

- Belongie et al. [5] achieved 0.63%.
- Keysers et al. [6] achieved 0.54% and 0.52% using K-NN.
- Kégl and Busa-Fekete [7] attained 0.87% using boosted stumps on Haar features.
- LeCun et al. [4] achieved 0.8%.
- Decoste and Schölkopf [8] reported results ranging from 0.56% to 0.68% using SVM.

This analysis highlights the historical landscape of OCR methodologies, emphasizing the dominance of CNNs while acknowledging the competitive performance of classical techniques.

5 Source Codes

- Source code with outputs for detailed EDA can be found at `https://colab.research.google.com/drive/13G8g_PAWwPGdI0mkybYq6zQziIjS7Ekd#scrollTo=kjE81uwuby8h`.
- Final Source code with outputs can be found at `https://colab.research.google.com/drive/13G8g_PAWwPGdI0mkybYq6zQziIjS7Ekd#scrollTo=kjE81uwuby8h`.

References

[1] Baldominos, Alejandro, et al. "A Survey of Handwritten Character Recognition with MNIST and EMNIST." Applied Sciences, vol. 9, no. 15, Jan. 2019, p. 3169. www.mdpi.com, https://doi.org/10.3390/app9153169.

[2] Tahir, Ammar, and Adil Pervaiz. "Hand Written Character Recognition Using SVM." Pacific International Journal, vol. 3, no. 2, June 2020, pp. 59–62. rclss.com, https://doi.org/10.55014/pij.v3i2.98.

[3] Alsaafin, Areej, and Ashraf Elnagar. "A Minimal Subset of Features Using Feature Selection for Handwritten Digit Recognition." Journal of Intelligent Learning Systems and Applications, vol. 9, no. 4, Sept. 2017, pp. 55–68. www.scirp.org, https://doi.org/10.4236/jilsa.2017.94006.

[4] Gope, Birjit, et al. "Handwritten Digits Identification Using Mnist Database Via Machine Learning Models." IOP Conference Series: Materials Science and Engineering, vol. 1022, no. 1, Jan. 2021, p. 012108. Institute of Physics, https://doi.org/10.1088/1757-899X/1022/1/012108.

[5] `https://stats.stackexchange.com/questions/22569/pca-and-proportion-of-variance-explained`.

[6] `https://www.datacamp.com/tutorial/introduction-t-sne`.

[7] `https://www.geeksforgeeks.org/os-walk-python/`.

[8] `https://github.com/OSSpk/Handwritten-Digits-Classification-Using-KNN-Multiclass_Perceptron-SVM`.

[9] `https://www.geeksforgeeks.org/ml-implementation-of-knn-classifier-using-sklearn/`.

[10] `https://towardsdatascience.com/hog-histogram-of-oriented-gradients-67ecd887675f`.

[11] `https://scikit-image.org/docs/stable/auto_examples/edges/plot_canny.html`.