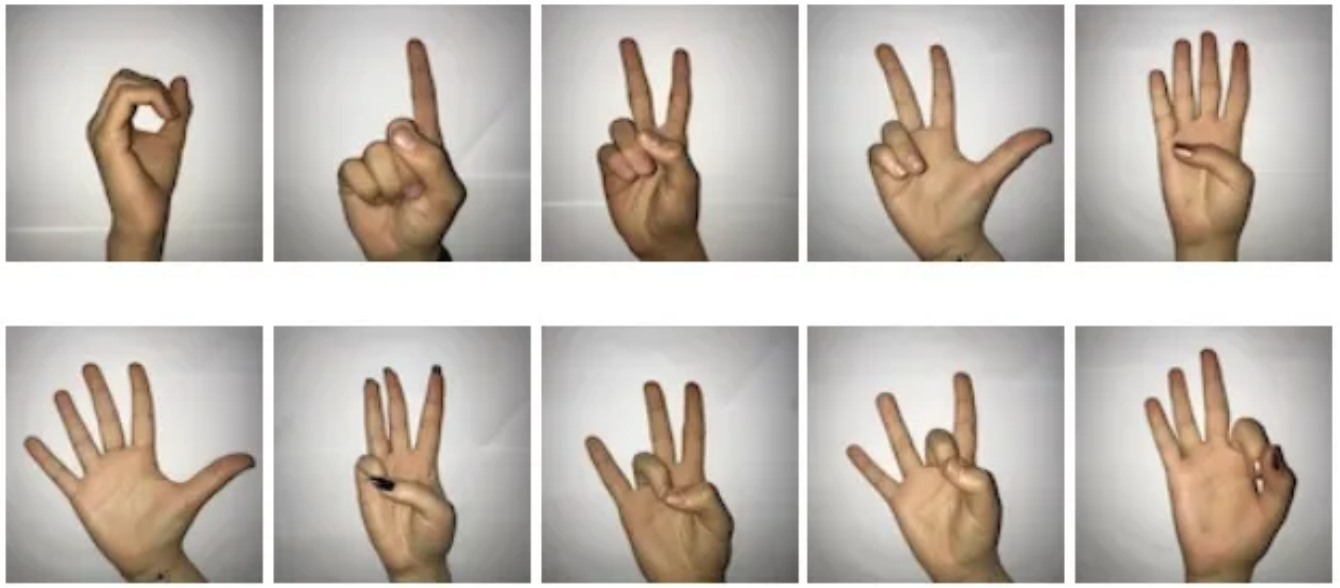
Identification of Digits from sign languages



José Santos, 98279

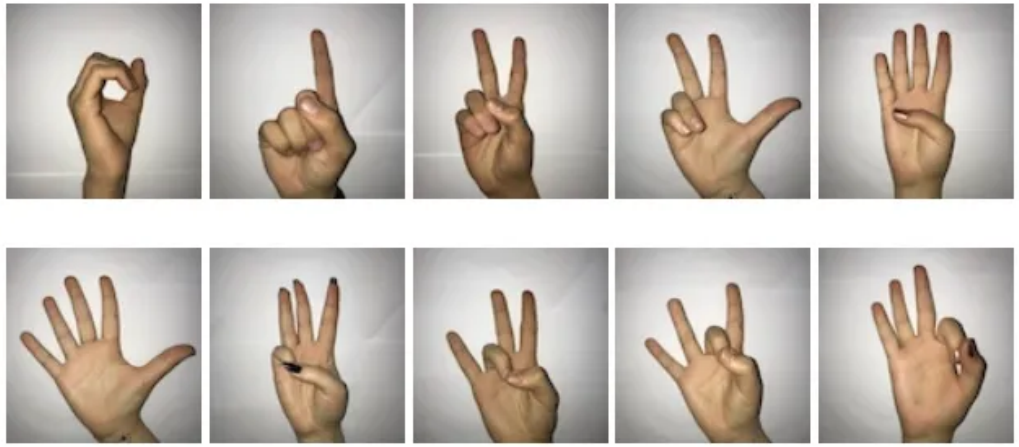
Henrique Sousa, 98324

Introduction

- Can be useful for:
  - Improving communication between hearing and non-hearing individuals;
  - New technologies for the deaf and hard-of-hearing community.
- Different models were trained with a dataset from Kaggle and their performance was compared.
- Studied impact of some preprocessing techniques.

Data

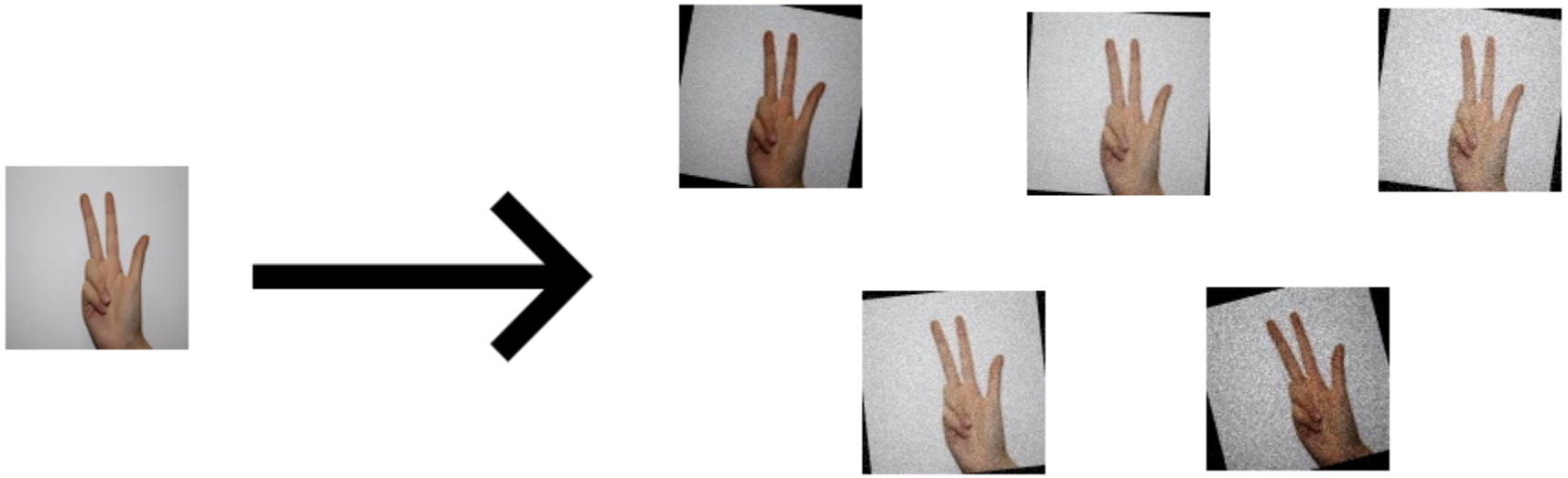
- Original dataset has 2062 images
- Includes all numbers from 0 to 9
- Balanced dataset with similar number of examples for each label



Data Preprocessing

Image Augmentation

- We created 5 variations of each image by randomnly:
  - Rotating from -20 to 20 degrees
  - Adding Gaussian noise
  - Changing the gamma contrast by a random factor



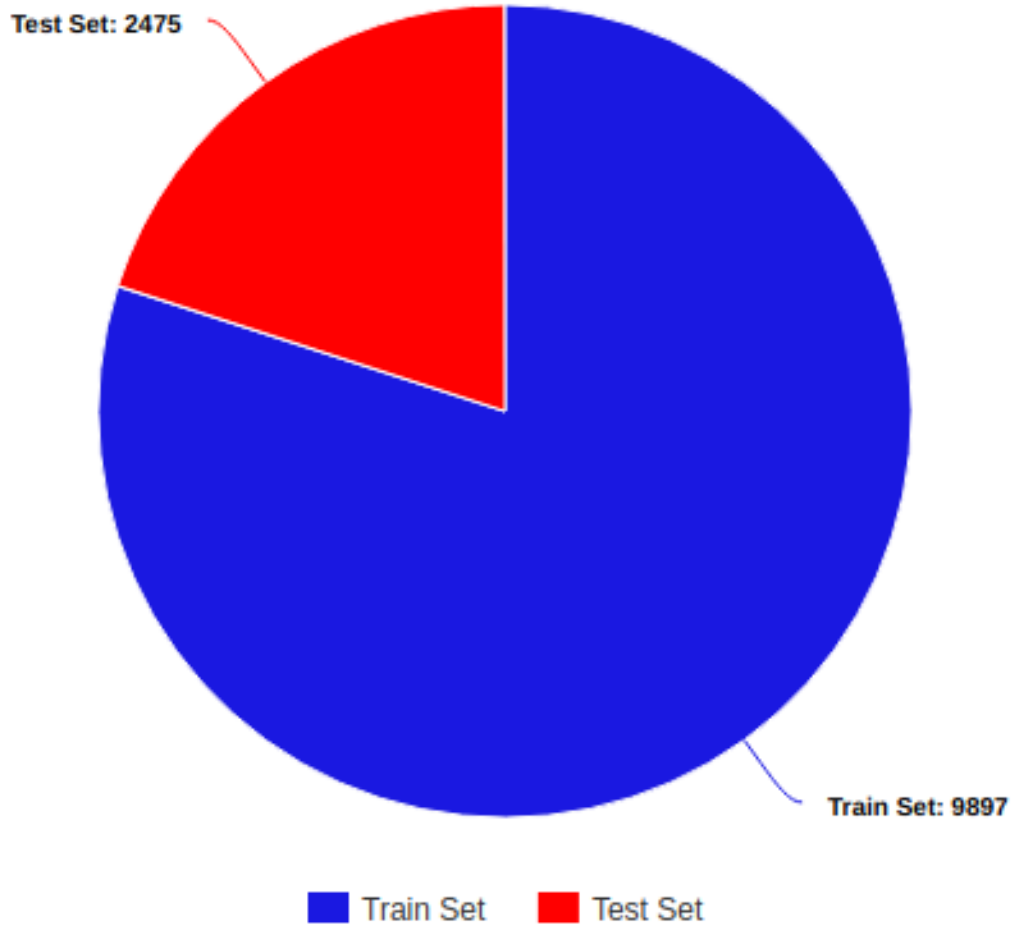
Data Preprocessing

Image processing

- Resize image to 50x50
- Convert it to grayscale
- Flatten the images

Training and Testing data

- Training data: 80% of the augmented dataset
- Testing data: 20% of the augmented dataset



Models

- Initial run with default parameters:

Model	Accuracy	F1 score
Multilayer Perceptron Classifier	0.092	0.015
Naive Bayes	0.502	0.506
Decision Tree Classifier	0.631	0.632
Logistic Regression	0.750	0.749
Random Forest Classifier	0.876	0.876
Support Vector Machines	0.888	0.888

Hyperparameter Tuning & Cross-Validation

We tried to find the optimal parameters for our case for the following Machine Learning models:

- Support Vector Machines
- Random Forest Classifier
- Logistic Regression
- Multilayer Perceptron Classifier

Multilayer Perceptron

- Initial Accuracy of 9.2% and F1 Score of 1.5%

Parameter	Values	Best Value
solver	[adam, lbfgs]	lbfgs
max_iter	[250, 500, 1000]	1000
hidden_layer_sizes	From 1 to 3 hidden layers	(256, 512, 128)
activation	[relu, tanh]	relu
alpha	[0.0001, 0.001, 0.01]	0.0001
learning_rate	[constant, invscaling, adaptive]	adaptive
learning_rate_init	[0.001, 0.01, 0.1]	0.001

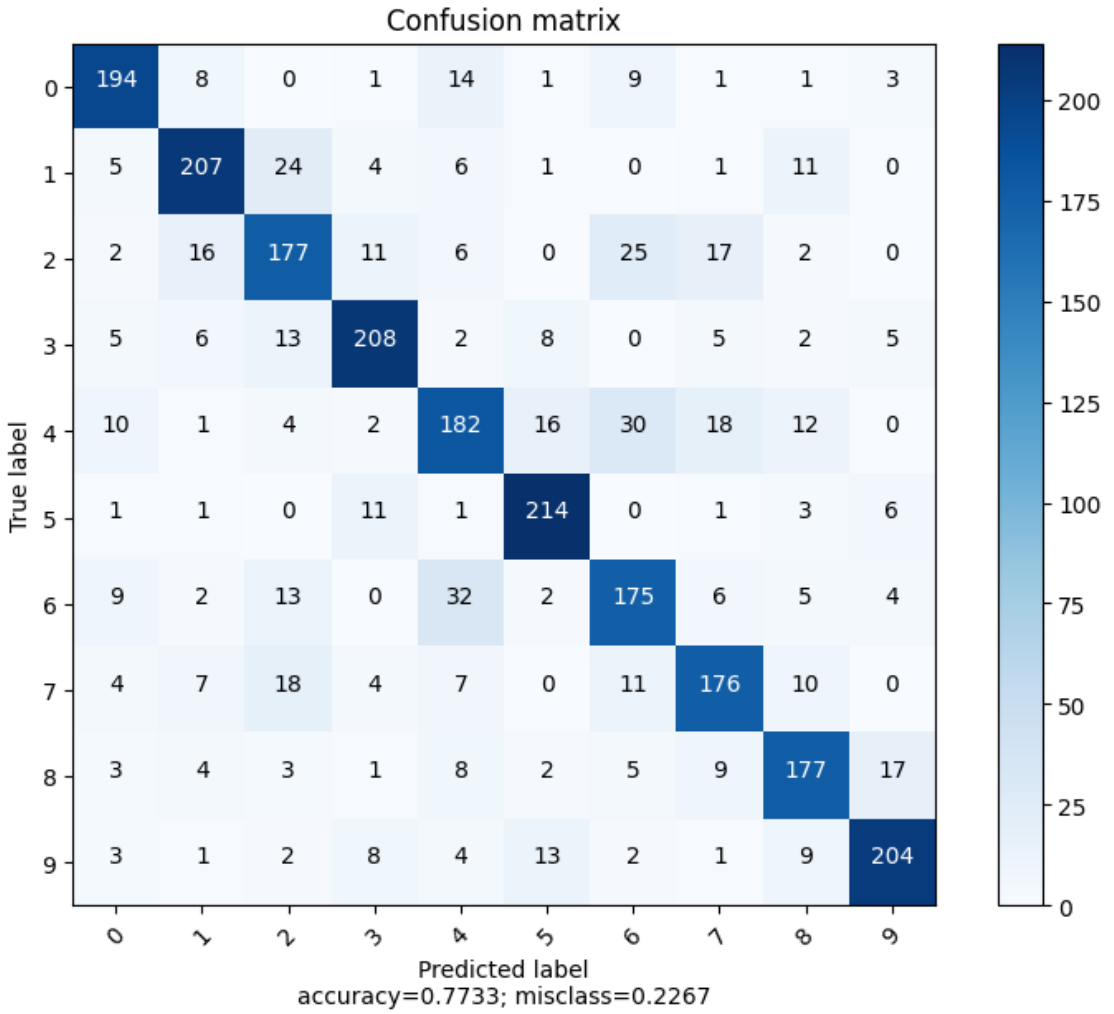
TABLE VI
MULTILAYER PERCEPTRON PARAMETERS

Multilayer Perceptron

- Final Accuracy and F1 Score of 0,77
- Increase of ~739.13%

Label	Precision	Recall	F1 Score
0	0.82	0.84	0.83
1	0.82	0.80	0.81
2	0.70	0.69	0.69
3	0.83	0.82	0.83
4	0.69	0.66	0.68
5	0.83	0.90	0.86
6	0.68	0.71	0.69
7	0.75	0.74	0.75
8	0.76	0.77	0.77
9	0.85	0.83	0.84

TABLE VII
MULTILAYER PERCEPTRON REPORT



Logistic Regression

- Initial Accuracy and F1 Score of 75%

Parameter	Values	Best Value
solver	[lbfgs]	lbfgs
max_iter	[100, 400, 800]	1000
C	[0.1, 1, 10, 100, 1000]	10
class_weight	[balanced]	balanced
penalty	[l2]	l2

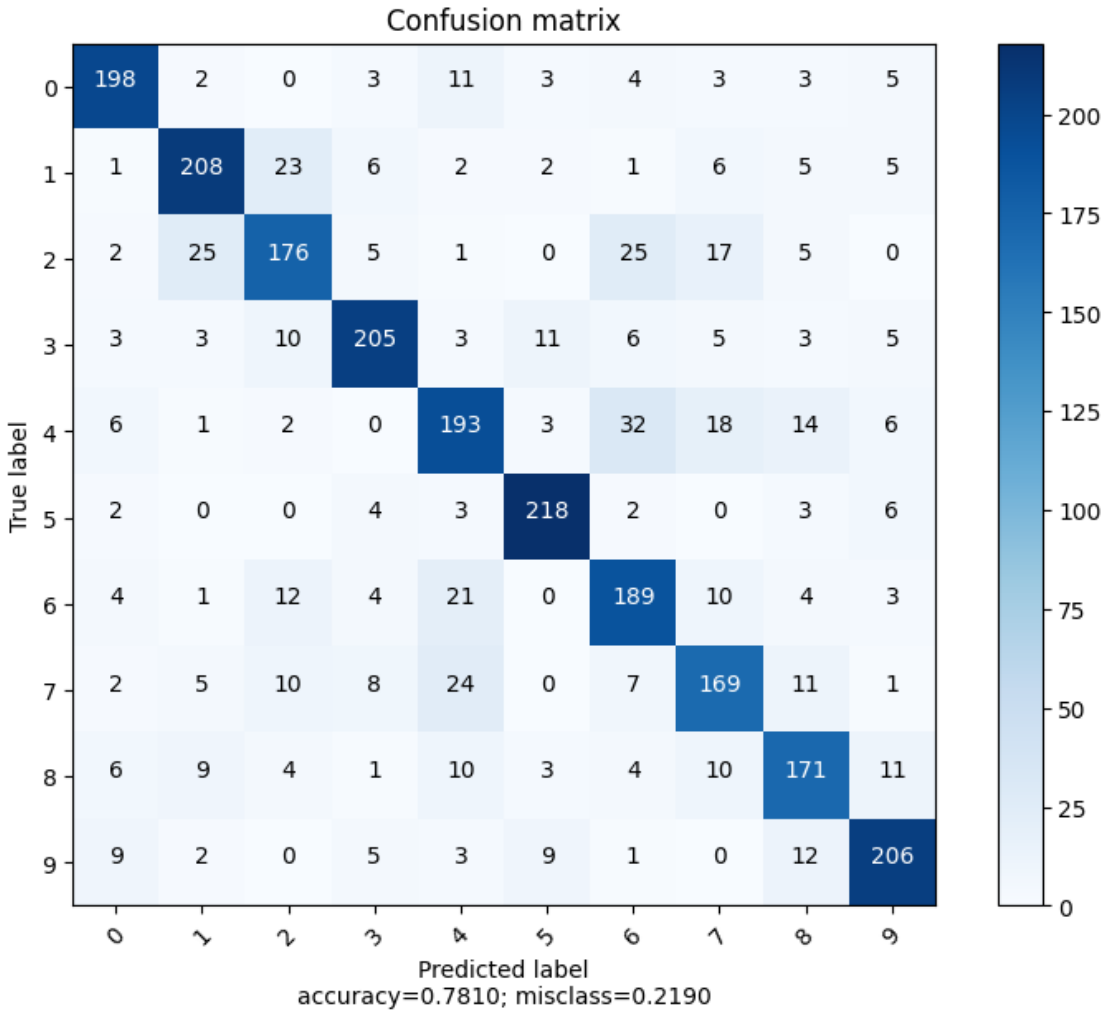
TABLE VIII
LOGISTIC REGRESSION PARAMETERS

Logistic Regression

- Final Accuracy and F1 Score of 0,78
- Increase of ~4%

Label	Precision	Recall	F1 Score
0	0.85	0.85	0.85
1	0.81	0.80	0.81
2	0.74	0.69	0.71
3	0.85	0.81	0.83
4	0.71	0.70	0.71
5	0.88	0.92	0.90
6	0.70	0.76	0.73
7	0.71	0.71	0.71
8	0.74	0.75	0.74
9	0.83	0.83	0.83

TABLE IX
LOGISTIC REGRESSION REPORT



Random Forest Classifier

- Initial Accuracy and F1 Score of 87,6%

Parameter	Values	Best Value
n_estimators	[50, 100, 200, 500]	500
criterion	[gini, entropy]	entropy
max_depth	[None, 5, 10, 20]	None
min_samples_split	[2, 5, 10]	2
min_samples_leaf	[1, 2, 4]	1
max_features	[auto, sqrt, log2]	auto

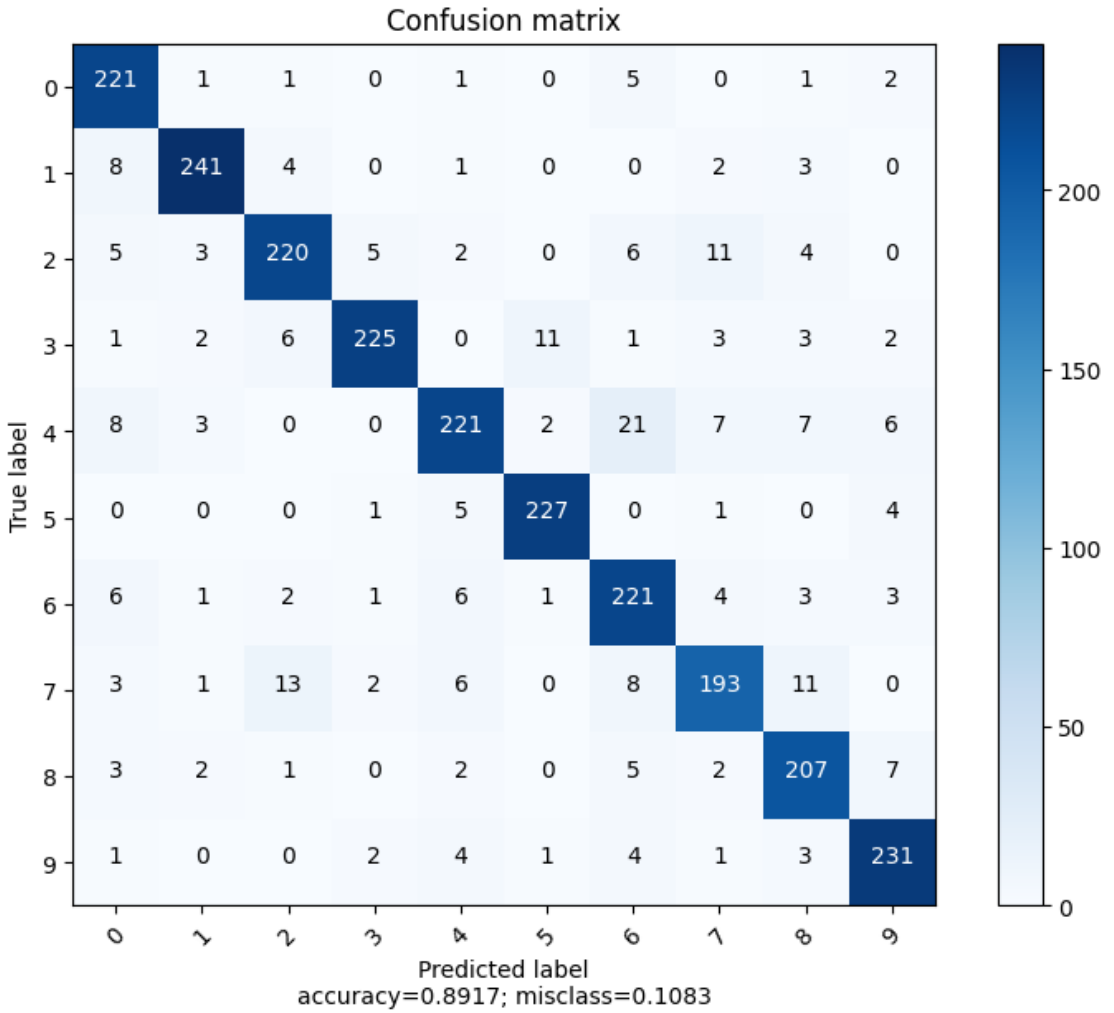
TABLE IV
RANDOM FOREST CLASSIFIER PARAMETERS

Random Forest Classifier

- Final Accuracy and F1 Score of 0.89
- Increase of ~1,7%

Label	Precision	Recall	F1 Score
0	0.86	0.95	0.91
1	0.95	0.93	0.94
2	0.89	0.86	0.87
3	0.95	0.89	0.92
4	0.89	0.80	0.85
5	0.94	0.95	0.95
6	0.82	0.89	0.85
7	0.86	0.81	0.84
8	0.86	0.90	0.88
9	0.91	0.94	0.92

TABLE V
RANDOM FOREST CLASSIFICATION REPORT



Support Vector Machines

- Initial Accuracy and F1 Score of 88,8%

Parameter	Values	Best Value
C	[0.001, 0.01, 0.1, 1, 10, 100, 1000]	100
kernel	[linear, poly, rbf, sigmoid]	rbf
degree	[2, 3, 4, 5]	2
gamma	[scale, auto]	scale

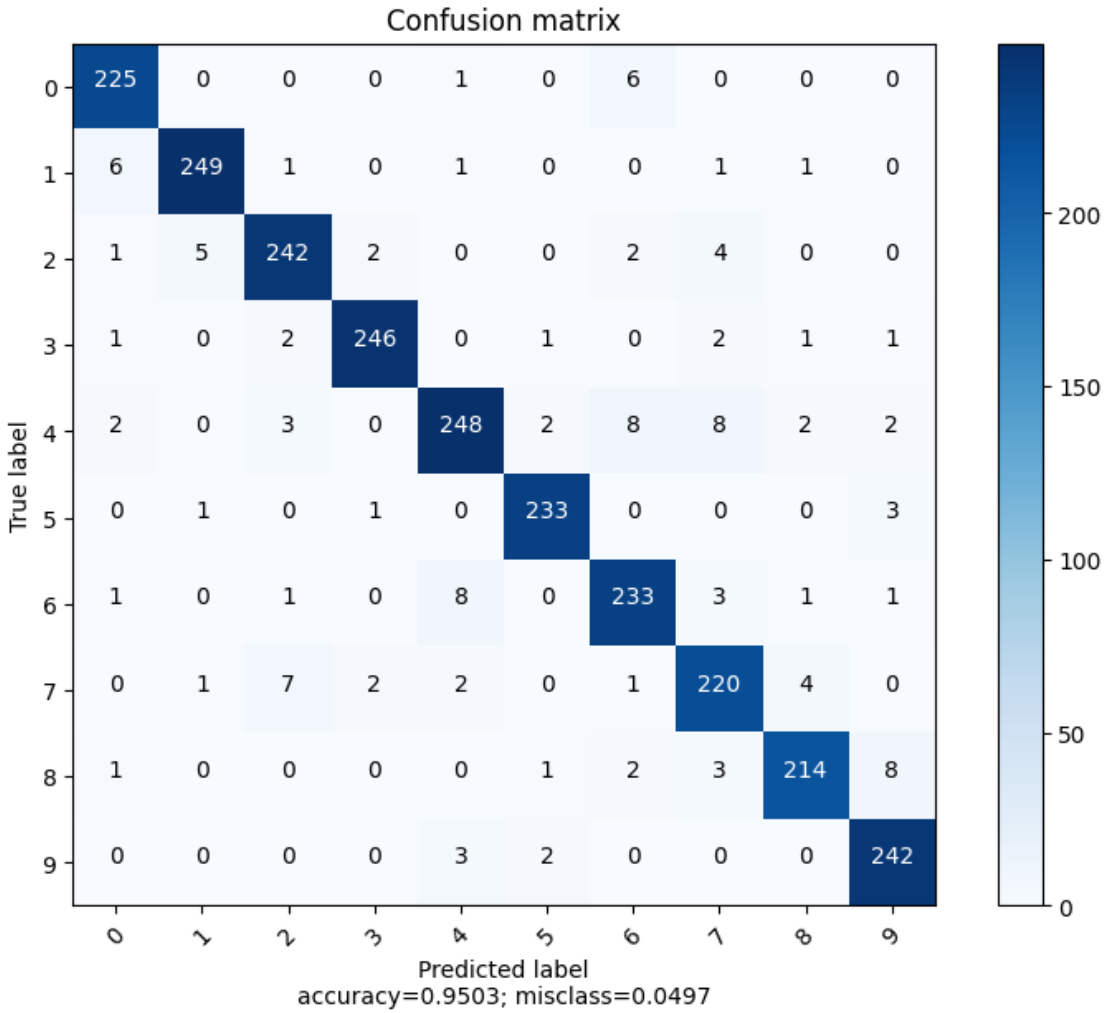
TABLE II
SVM PARAMETERS

Support Vector Machines

- Final Accuracy and F1 Score of 0.95
- Increase of ~6,7%

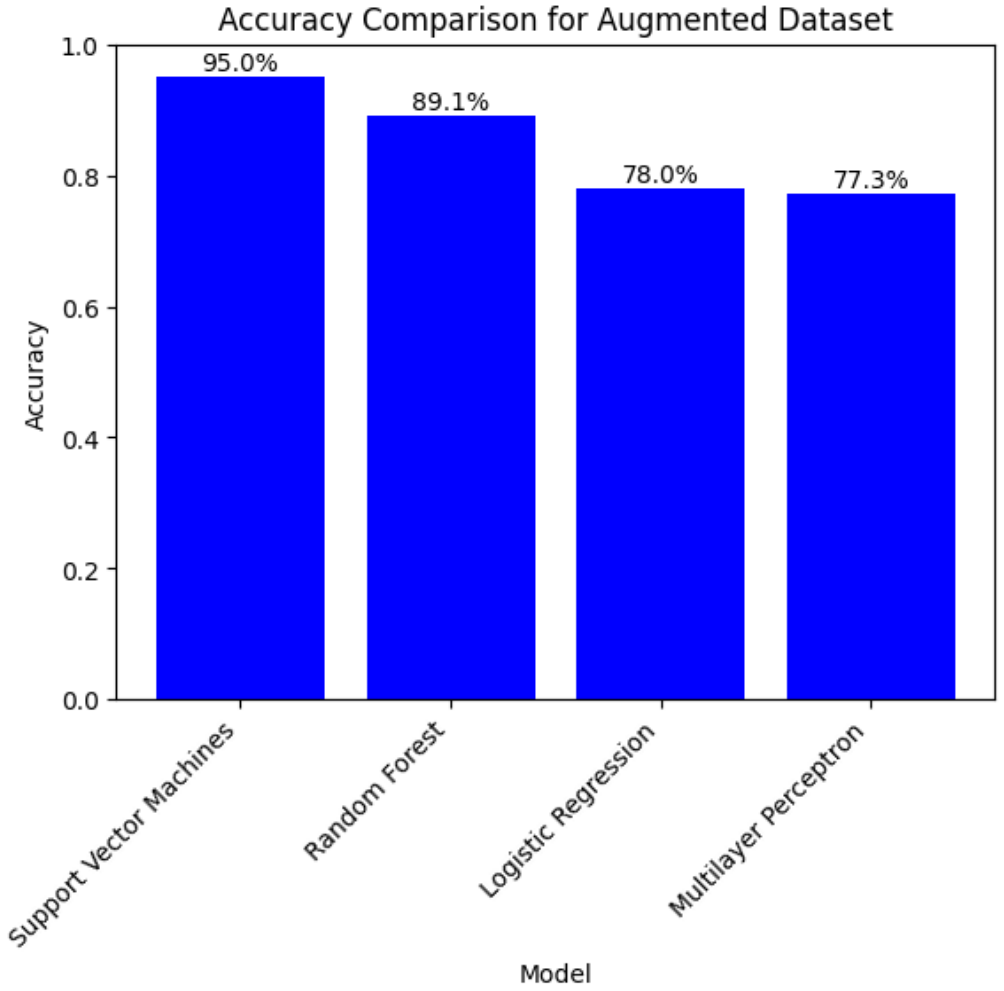
Label	Precision	Recall	F1 Score
0	0.95	0.97	0.96
1	0.97	0.96	0.97
2	0.95	0.95	0.95
3	0.98	0.97	0.97
4	0.94	0.90	0.92
5	0.97	0.98	0.98
6	0.92	0.94	0.93
7	0.91	0.93	0.92
8	0.96	0.93	0.95
9	0.94	0.98	0.96

TABLE III
SVM CLASSIFICATION REPORT



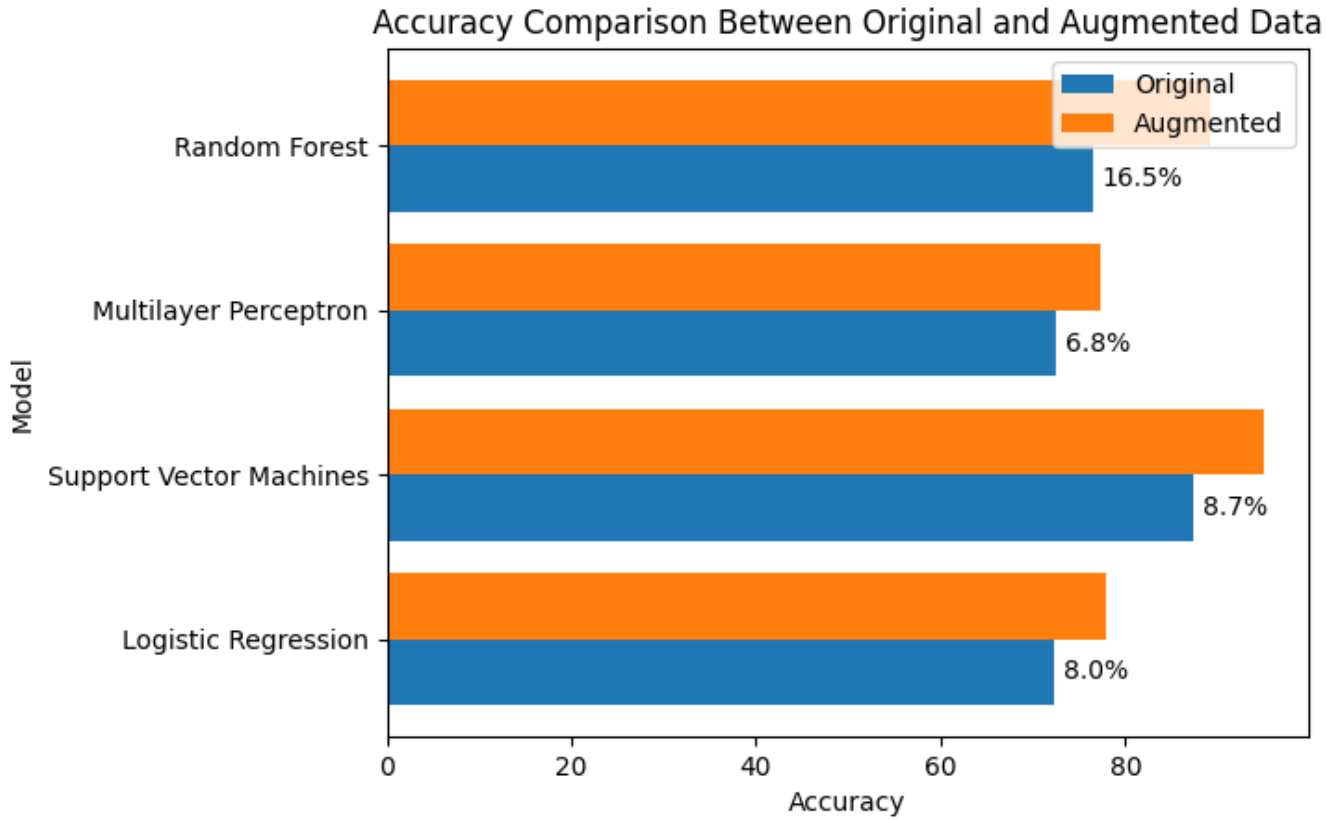
Model Comparison

- Support Vector Machines achieved highest accuracy with 95.0%
- Random Forest followed with 89.1%
- Logistic Regression achieved 78.0%
- Multilayer Perceptron achieved 77.3%



Original vs Augmented Dataset

- In the graph below we can see the difference in performance by using the original dataset and the augmented one



Novelty and Contribution

- We focused on identifying all 10 digits while most notebooks we found only focused on two digits
  - Made it difficult to compare performances
- Pavan's notebook on automated hyperparameter tuning helped us decide the better way to find the optimal parameters
- Aleju's imgaug presented a simple and easy-to-use python library for generating additional images

Conclusion

- Support Vector Machine was the best performing model for this task
- We were able to achieve a good performance with our changes
- Most important aspect of these tests is the use of good data
- Image augmentation is extremely important for these kind of problems when dealing with small datasets

Questions?