Optimizing Facial Emotion Recognition

Edmond Anderson, Suliah Apatira, Jasmine Harris
CMSE 831: Computational Optimization
Longxiu Huang

Facial Image Classification

Facial emotion image classification is an AI-driven field focused on automatically recognizing and categorizing emotions expressed in human faces using advanced algorithms and deep learning models.

This technology has the potential to revolutionize human-computer interaction, mental health applications, and our understanding of emotions across various contexts.

Strategy

1. Explore Data
2. Preprocessing
3. Dimensionality Reduction
4. Model Selection and Tuning
5. Performance Evaluation

01 Exploratory Data Analysis

About the Dataset

- **6 Expressions:** Surprise, Anger, Fear, Happy, Neutral, Sad
- **Dimensions:** 48 x 48 pixel grayscale images

35, 340 Images

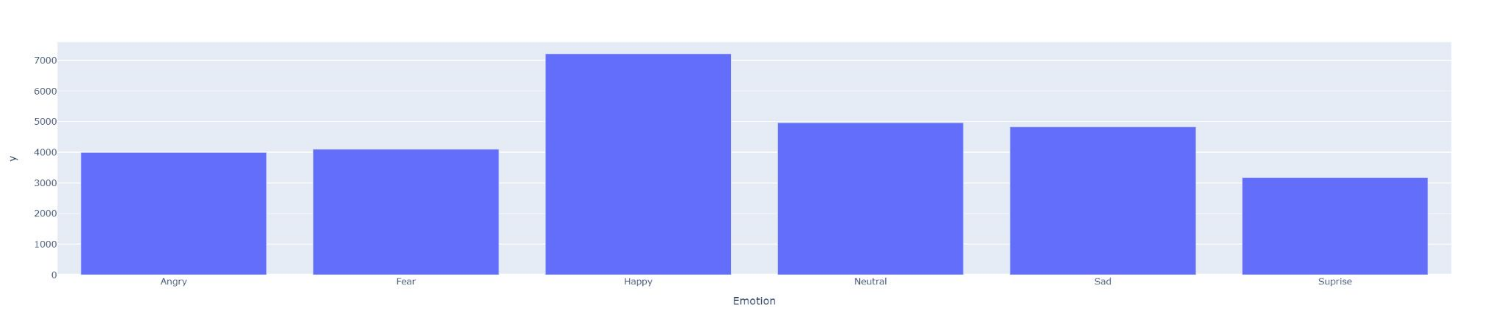
Train Set: 28,273

Test Set: 7, 067

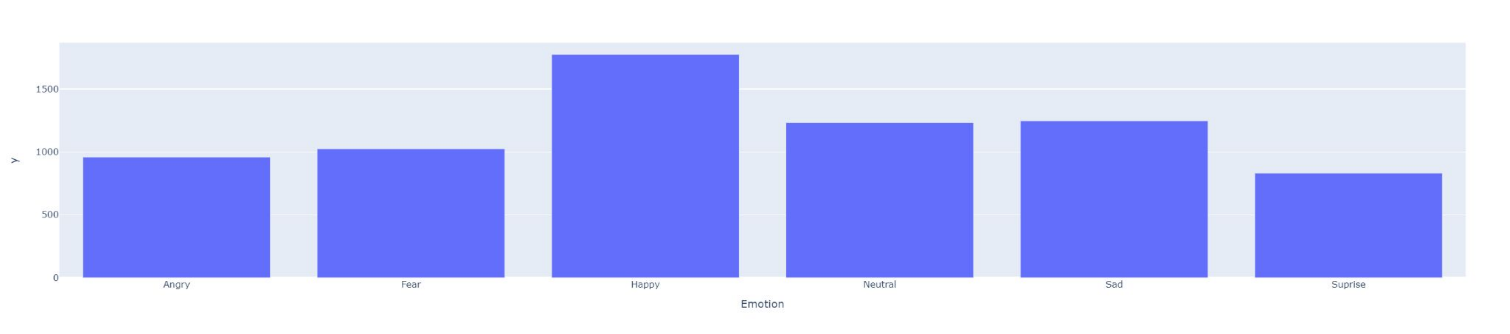
Sample Images

Emotion Breakdown

80% Train



20% Test



02 Preprocessing

Preprocessing

- 30% augmented data added
- Height randomly shifted by up to 20%
- Width randomly shifted by up to 20%
- Images randomly flipped horizontally
- Images randomly rotated left/right by up to 20 degrees
- RGB values normalized (0-255) -> (0-1)


Angry Face
Suprise Face
Neutral Face
Fear Face
Happy Face
Sad Face
Fear Face
Fear Face
Suprise Face
Neutral Face
Sad Face
Happy Face
Neutral Face
Neutral Face
Happy Face

Train Set:
(28273, 48, 48)

Augmented Set:
(36754, 48,48)

Scaled Set
(36754, 2304)

04
Dimensionality Reduction

\

Dimensionality Reduction

Principal Component Analysis

A mathematical technique to reduce the dimensionality of data. It works on the principal of factoring matrices to extract the principal pattern of a linear system

Linear Discriminant Analysis

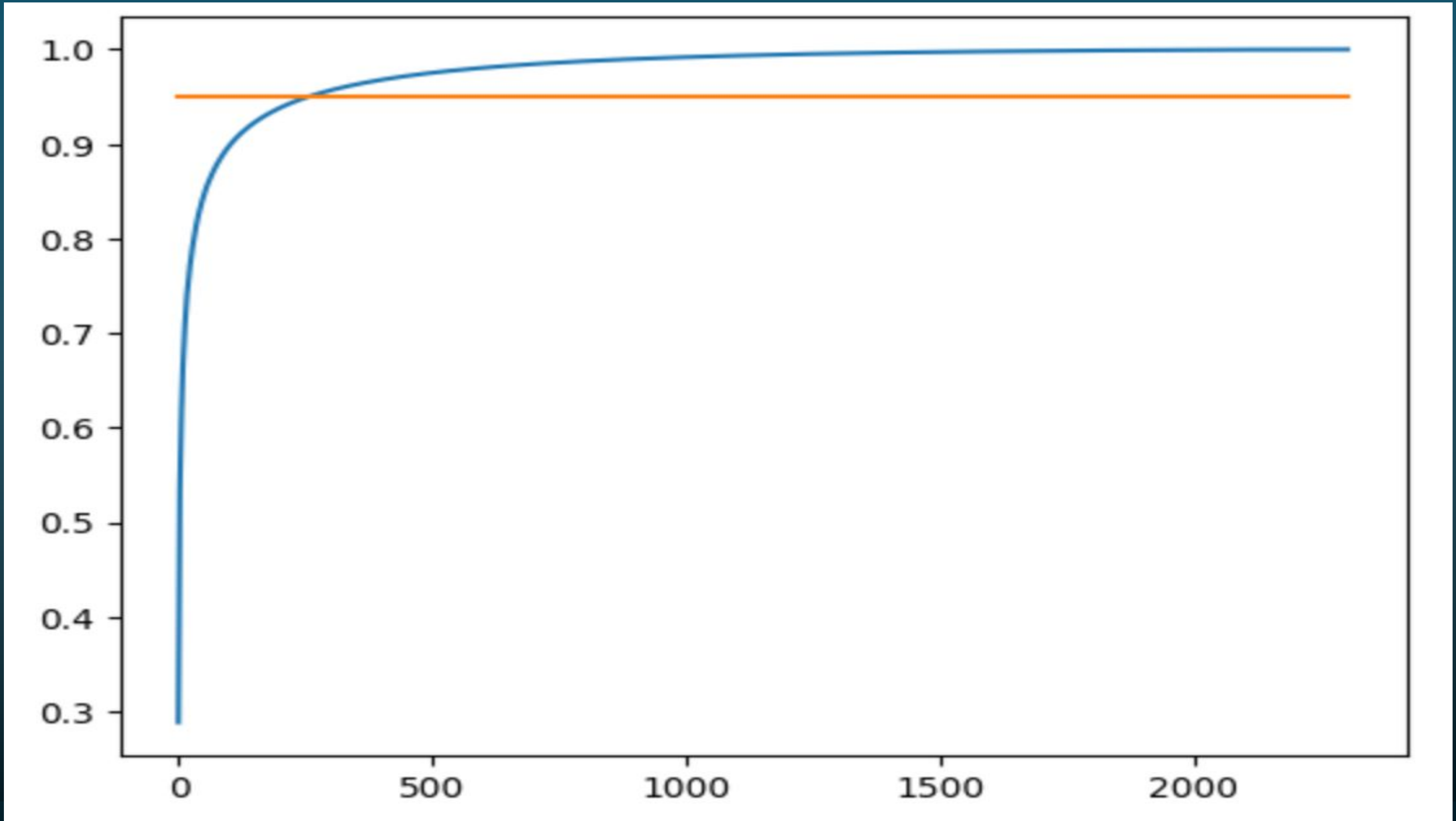
This classical supervised dimensionality reduction method seeks an optimal linear transformation that maps the original data to a low-dimensional space.

Scaled Set:
(36,754, 2304)

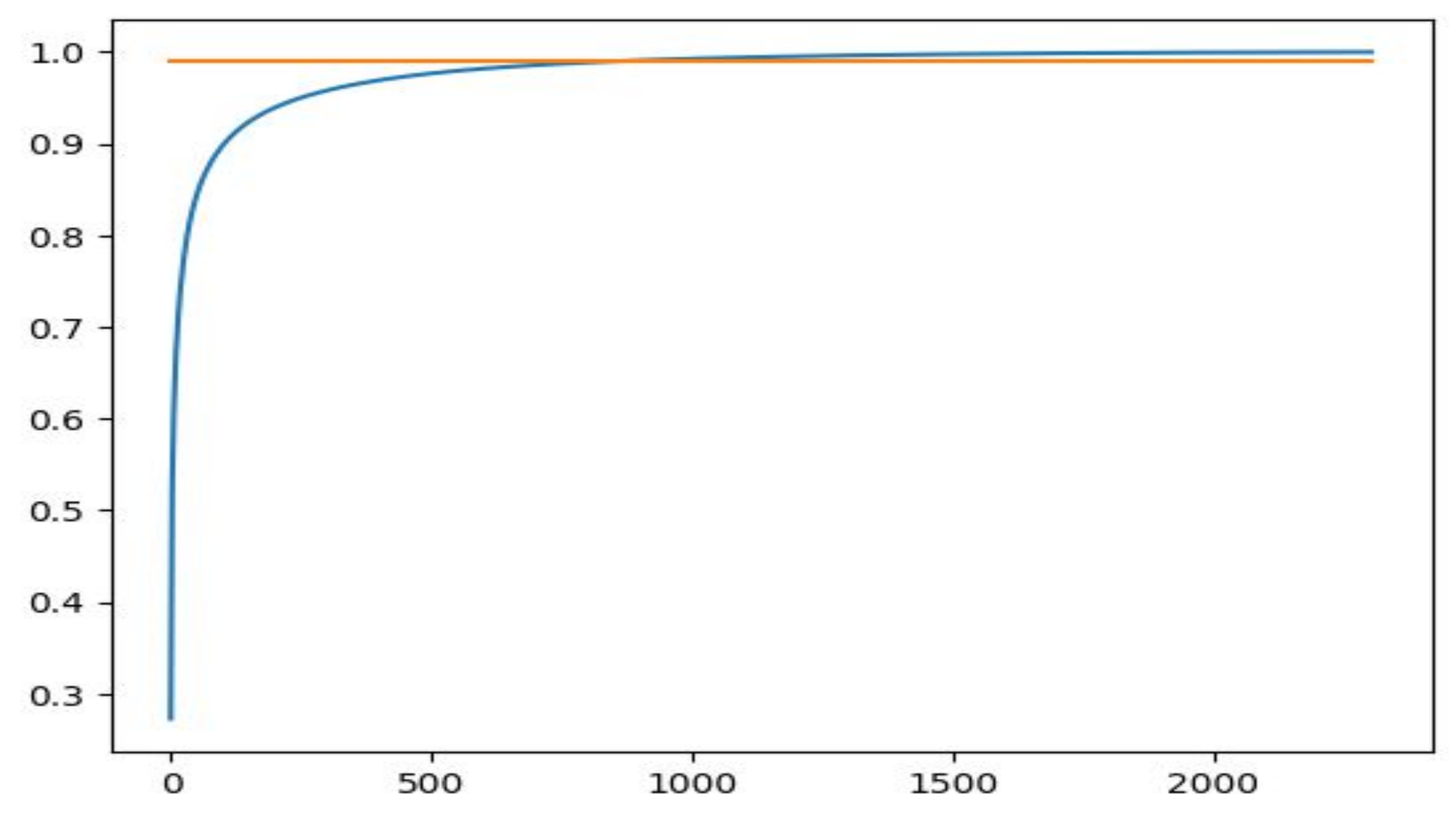
After PCA:
(36,754, 856)

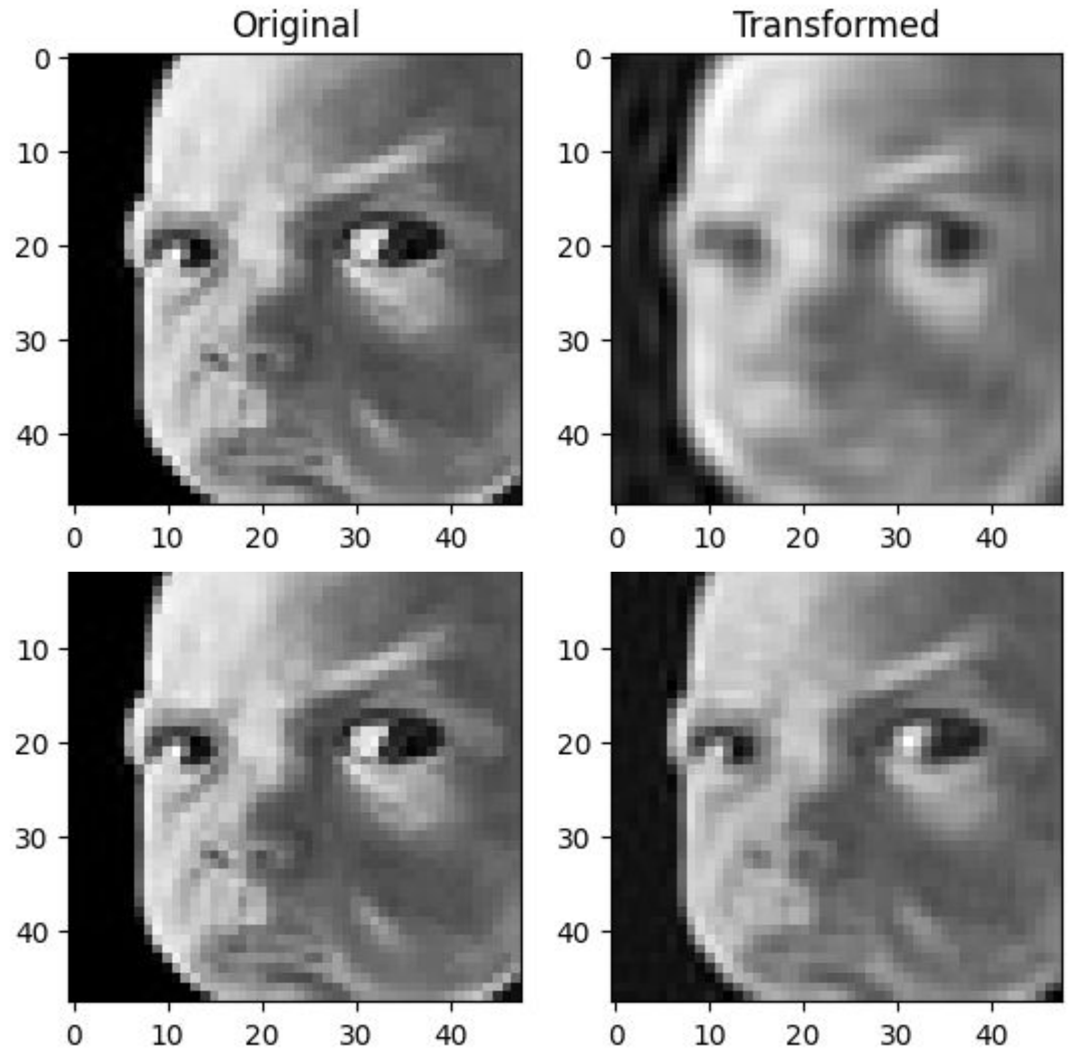
After LDA
(36754, 5)

PCA



PCA




Original
Transformed

LDA

Between Class Variance

$$S_b = \sum_{i=1}^{g} N_i(\bar{x}_i - \bar{x})(\bar{x}_i - \bar{x})^T$$

Within Class Variance

$$S_w = \sum_{i=1}^{g} (N_i - 1)S_i = \sum_{i=1}^{g}\sum_{j=1}^{N_i} (x_{i,j} - \bar{x}_i)(x_{i,j} - \bar{x}_i)^T$$

Lower Dimensional Space

$$P_{lda} = \arg\max_{P} \frac{\left|P^T S_b P\right|}{\left|P^T S_w P\right|}$$

05
Model Selection and Tuning

\

Stochastic Gradient Descent

Model Overview

$$Q(w) = \frac{1}{n}\sum_{i=1}^{n} Q_i(w),$$

Pros:

- Efficiency and speed
- Convergence rate
- Regularization
- Memory efficiency

Cons:

- Noisy updates
- Varied convergence
- Learning rate running
- Sensitivity to initial conditions

Hyperparameters Tested:

- '***alpha***': [0.0001, **0.001**, 0.01],
- '***loss***': ['hinge', **'log_loss'**, 'modified_huber'],
- '***penalty***': ['l1', **'l2'**, 'elasticnet'],
- '***max_iter***': [1000, 2000, **3000**],
- '***tol***': [1e-3, 1e-4, **1e-5**],

Best Model:

Validation Score: 0.3795749497731347
Test Score: 0.36521862176312436

Naive Bayes

Model Overview

$$(x = v \mid C_k) = \frac{1}{\sqrt{2\pi\sigma_k^2}} e^{-\frac{(v-\mu_k)^2}{2\sigma_k^2}}$$

Pros:

- Simplicity and speed
- Computational efficiency
- Scalability

Cons:

- Assumes feature independence
- Sensitive
- Often too simple for complex relations

Hyperparameters Tested:

- ***'var_smoothing':*** [**5e-9**,1e-9,5e-8,1e-8,5e-7,1e-7,5e-6, 1e-6,5e-5,1e-5]

Best Model:

Best Score: 0.37777
Test Score: 0.36861

MultiLayer Perceptron

Model Overview

Neural network that learns the relationship between linear and non-linear data.
Layers - Input, Hidden, and Output

$$f(x) = f^{(3)}(f^{(2)}(f^{(1)}(x)))$$

Pros:

- Fully Connected Network
- Can handle both structured and unstructured data

Cons:

- Prone to overfitting
- Sensitive to noise data

Hyperparameters Tested:

- ***'hidden_layer_sizes':*** [**(50,)**, (100,), (50,50), (100,50)]**,**
- ***'activation':*** [**'relu'**, 'tanh']**,**
- ***'solver':*** ['adam', 'sgd']***,***
- ***'alpha':*** [0.0001, 0.001, 0.01]***,***

Best Model:

Validation Score: 0.38254
Test Score: 0.36819

Support Vector Machine

Model Overview

Supervised machine learning problem that finds a hyperplane to separate the classes.

$$\forall n: \left| y_n - (x_n'\beta + b) \right| \leq \varepsilon$$

Pros:

- Memory Efficient
- Can be used for both linear and non-linear classification
- Robust to overfitting

Cons:

- Computational complexity
- Sensitive to noise
- Memory Usage

Hyperparameters Tested:

- ***'C'***: [0.1, .5, 1, 10],
- ***'kernel'***: ['linear', 'rbf']
- ***'gamma':*** ['scale', 'auto']

Best Model:

Validation Score: 0.41289
Test Score: 0.42075

VGG19

Model Overview

VGG-19 is a convolutional neural network that is 19 layers deep.

Pros:

- Great Accuracy
- Simple to implement and understand
- Great at extracting rich features from images

Cons:

- Can be computationally expensive

Hyperparameters Tested:

- ***'units':*** [min_value=256, max_value=1024, step=32, (300)],
- ***'dropout':*** [0.1, 0.2, 0.3, '0.4', 0.5'],
- ***'solver':*** ['adam', 'sgd'],
- ***'learning_rate':*** [min_value=1e-6, max_value=1e-3 (6.364e-05)],

Best Model:

Validation Score: 0.6549
Test Score: 0.66

ResNet50

Model Overview

ResNet-50 is a 50-layer convolutional neural network (48 convolutional layers, one MaxPool layer, and one average pool layer)

Pros:

- Depth Handling with residuals
- Ability to use pretrained weights

Cons:

- Computationally expensive
- Sensitive to noisy data

Hyperparameters Tested:

- ***'units':*** [min_value=256, max_value=1024, step=32, (1024)],
- ***'dropout':*** [0.1, 0.2, 0.3, '0.4', 0.5'],
- ***'solver':*** ['adam', 'sgd'],
- ***'learning_rate':*** [min_value=1e-6, max_value=1e-3 (9.8648e-05)],

Best Model:

Validation Score: 0.6194
Test Score: 0.62

Results

MODELS	ACCURACY
SGD	36.5%
NB	36.9%
MLP	36.8%
SVM	42%
CNN, VGG19	67%
CNN, ResNet50	62%

Summary and Conclusions

- **Data** — The dataset we obtained from Kaggle contained 35, 340 images
- **Preprocessing** — We scaled our data for normalization
- **Dimensionality** — We reduced our data to 5 dimensions for computational efficiency
- **Modeling** — We used 5 modeling techniques
- **Results** — The best model was.....VGG19

Mislabeled Dataset

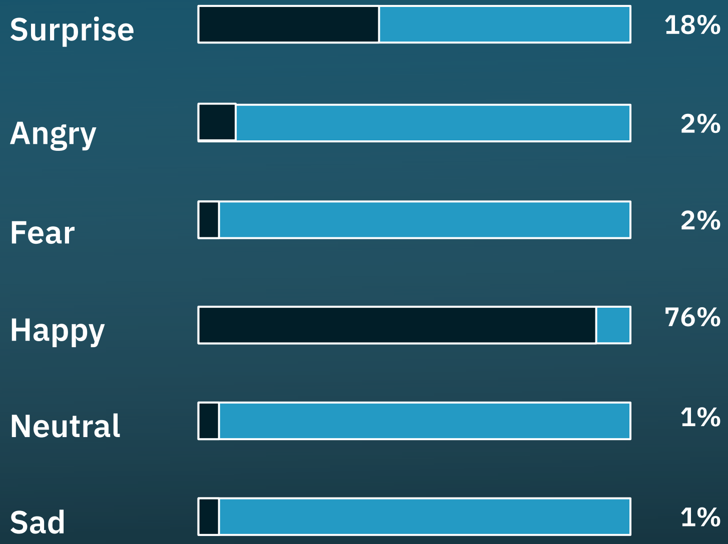
Time Constraints

Limitations

Computational Resources

Keras Libraries

Future Applications - Build a Website



Thanks!

Any questions?

References

- Facial Recognition Dataset (kaggle.com)
- Image Detection Using the VGG-19 Convolutional Neural Network | by Melisa Bardhi | MLearning.ai | Medium
- Programming Image Classification with Machine Learning (kili-technology.com)
- Dimensionality Reduction Techniques | Python (analyticsvidhya.com)
- Image classification | TensorFlow Core
- fit_transform(), fit(), transform() in Scikit-Learn | Uses & Differences (analyticsvidhya.com)
- Introduction to PCA: Image Compression example | Kaggle
- Building a Topic Modelling for Images using LDA and Transfer Learning | by Saikumar Jagadeeswaran | Analytics Vidhya | Medium

SCALED

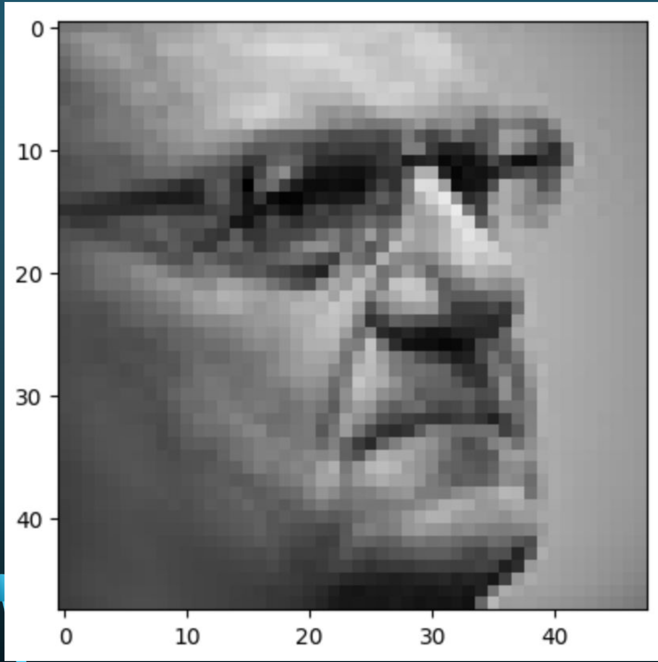
MODELS	TIME	ACCURACY
SGD	3 min 25 sec	29%
MLP	6 mins 7 sec	42%
RF	2 mins 25 sec	47%

SCALED - LDA

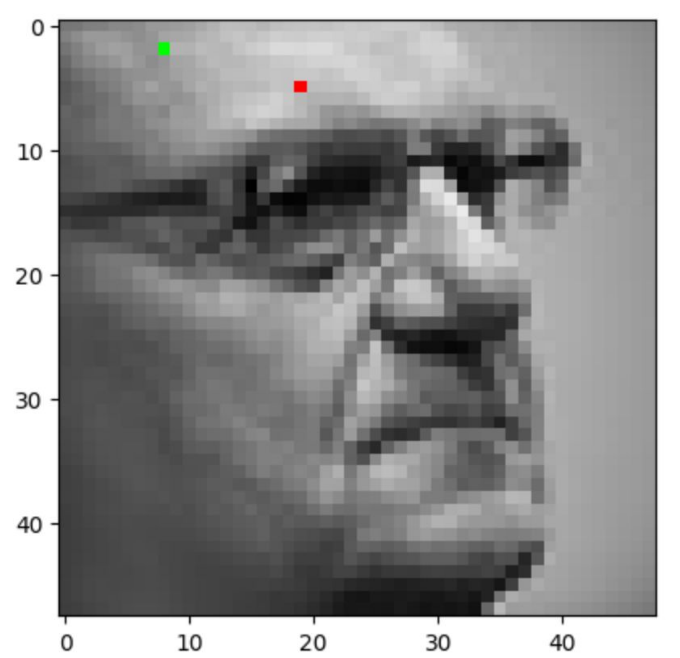
MODELS	TIME	ACCURACY
SGD	445 ms	65%
MLP	21.7 sec	66%
SVM	1 min 48 sec	67%
RF	9.69 sec	64%

```
plt.imshow(X_train[0])
X_train[0,0,0]
```

```
array([0.46666667, 0.46666667, 0.46666667], dtype=float32)
```




```
X_train[0,5,19] = [1,0,0]
X_train[0,2,8] = [0,1,0]
plt.imshow(X_train[0])
```



Limitations & Future Work

- Mislabeled Dataset
- Computational Resource Limitations
- Time Constraints
- Keras Libraries (

ORIGINAL

MODELS	TIME	ACCURACY
NB	12 mins	28%
MLP	8 mins 17 sec	37%
RF	4 mins	46%
NB	TIME	ACCURACY
SVM	422 ms	61%
VGG19	18.2 sec	67%
ResNet50	1 min 36 sec	67%
RF	9.82 sec	63%

Random Forest

Model Overview

$$RFfi_i = \frac{\sum_{j \in all\ trees} normfi_{ij}}{T}$$

Pros:

- High accuracy
- Robust to overfitting
- Feature importance
- Parallelization

Cons:

- Model interpretability
- Computational complexity
- Memory usage
- Biased toward dominant classes

Hyperparameters Tested:

- ***'n_estimators'*:** [100, 200, 300]
- ***'max_depth'*:** [None, 5, 10, 20]
- ***'min_samples_split'*:** [2, 5, 10]
- ***'min_samples_leaf'*:** [1, 2, 4]

Best Model:

Validation Score:
Test Score:

KERAS

Sample Images

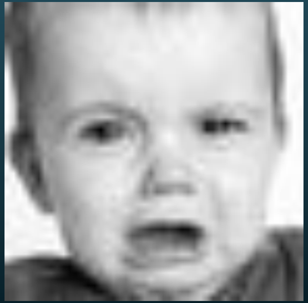
Angry

Fear

Happy

Neutral

Sad



Surprise