A face verification system

acq24bt

University of Sheffield

Abstract

Face verification is a classification task that detects whether the images are from the same or different person. This experiment create a machine learning model that try to exceed baseline accuracy. The system uses image augmentation, feature engineering, and preprocessing to optimise input data. Models, including Support Vector Machine (SVM), Random Forest, and Stacking Classifier, are trained using hyper-parameter tuning. Feature engineering, such as pixel and Euclidean distance calculations, and dimensionality reduction are crucial for performance improvement. The final model achieves 67.7% accuracy, outperforming the baseline by 11.4%, proving the effectiveness of this method.

1. Introduction

Face verification system identifies whether two images of human faces are from the same individual. The project aims to build a machine learning model that exceeds the accuracy of the baseline model provided, which is 56.3%. This involves image augmentation, feature engineering, data preprocessing, and model optimisation to create a robust model. In the provided dataset, each sample consists of sets of images representing an image pair of an individual. The challenges include splitting the image, creating more samples, extracting relationships between the images, and hyper-parameter tuning of the model.

This report describes the pipeline and methodology for solving the face verification problem. Image augmentations are used to increase training variations and sample size. Next, feature engineering is applied to determine the relationships of image pairs at both the pixel level and overall image scale. Data preprocessing methods, such as standardisation and PCA, are applied to optimise input features. Machine learning models, including Support Vector Machine (SVM), Random Forest, and Stacking Classifier, are used, and hyper-parameters are tuned for optimal results. The report concludes with an analysis of the results, the implementation, and suggestions for future improvements.

2. System Description

The system includes multiple steps. Data is imported from the 'train.joblib' file, which consists of two datasets. The image dataset contains 2,200 image pairs in 5,828-column vectors for each image, while the second dataset contains the corresponding labels of 1 and 0, determining whether the images are of the same or a different person. To increase variations and sample size, the image pairs are split two images, with identical augmentations applied to both images in each pair.

The images are then augmented using four different methods. First, horizontal flip, which maintains the image structure while adding variation. Second, a 15° rotation is applied to mimic misaligned images. Third, contrast adjustment is used to highlight differences in tone. Finally, brightness adjustment is applied to enhance bright spots in the images. These augmentations are selected with the goal of preserving overall details. The augmentations are added to the training data, as shown in Table 1. A total of 1,000 augmented samples are added to the training dataset to maintain the structure of the unaugmented samples.

Table 1: *Number of images being augmented*

Augmentations	Number of Samples
Horizontal Flip	200
Rotation	200
Contrast	300
Brightness	300

Feature engineering is applied to the augmented images. Absolute pixel-wise difference, $D_{\text{pixel}} = |I_1 - I_2|$ is applied to the image pairs. The images then go through Euclidean distance calculation, $D_{\text{euclidean}} = \sqrt{\sum_{i=1}^{n}(I_{1i} - I_{2i})^2}$, where I represents the image vector. This calculates the distance of the corresponding pixels and the overall distance of the image, determining the relationships and resulting in a feature vector. Standardisation and PCA are applied to the feature set to reduce dimensionality to 50 for model input.

The final stage involves training the model to classify the images. The system uses two models: Support Vector Machine (SVM) and Random Forest, with GridSearchCV used for hyper-parameter tuning and cross-validation. The SVM is tuned for regularisation strength (C): $[0.1, 1, 10, 100, 1000]$ and gamma $[0.001, 0.01, 0.1, 1]$, while using the RBF kernel. Random Forest is tuned with n_estimator: $[100, 200]$, max_depth: $[4, 5]$, min_samples_split and min_samples_leaf are both $[10, 15, 20]$.

Once the best hyper-parameters are determined for both models, a Stacking Classifier is used to combine their performance with the best hyper-parameters where Logistic Regression is used as a meta-estimator. The pipeline is created to handle feature engineering, standardisation, PCA, and model selection. The model and pipeline are then saved into a joblib file named 'model.joblib' to use in an evaluation task.

3. Experiments

The images are randomly selected and augmented, with augmentations as follows: horizontal flip, 15° rotation, adjusted contrast, adjusted brightness. These adjustments preserve the image style while adding important details, such as increased white spots, variations in skin tones, and mimic misaligned images. Other augmentations, such as vertical flip, blur, and inverse colour were excluded as they transformed the images unrealistically. The dataset size was increased by 1,000 samples to avoid overwhelming the original training dataset.

Feature engineering of absolute pixel distance and Euclidean distance is applied to the augmented images to understand the relationships and better represent the data in the model input. After feature engineering, the data is prepro-

cessed using standardisation and PCA dimensionality reduction to 50 components for improve generalisation and performance. The preprocessed feature dataset is tested on four models with default hyper-parameters using GridSearchCV with five-fold cross-validation. The models include SVM, Random Forest, MLPClassifier, and K-Nearest Neighbour (KNN). The results are presented in Table 2.

Table 2: *Results on default models*

Model	Training Accuracy	CV Accuracy
SVM	61.12%	52.87%
Random Forest	100%	63.82%
MLPClassifier	72.94%	57.53%
KNN	66.91%	52.81%

Due to the randomness of the augmentation in the training data, the MLPClassifier's training and cross-validation accuracy fluctuates significantly, ranging from 40% to 70% when run on the newly augmented dataset. As a result, this model is excluded.

Random Forest is observed to be highly overfitted, while SVM and KNN provide poor results. Therefore, Hyper-parameter tuning is required. SVM is tuned on regularisation strength to fine-tune the convergence of the model for improved results. Random Forest is tuned by increasing the minimum sample leaf and sample split values, as well as limiting the maximum depth to control overfitting. KNN is tuned by adjusting numbers of neighbours and weight configurations for an optimal performance. The hyper-parameters used are detailed in Table 3.

Table 3: *Table 3: Models Hyper-parameters*

Model	Hyper-parameters	values
SVM	C	0.1, 1, 10, 100, 1000
	gamma	0.001, 0.01, 0.1, 1
Random forest	n_estimator	100, 200
	max_dept	4,5
	min_sample_split	10, 15, 20
	min_sample_leaf	10, 15, 20
KNN	n_neighours	1, 2, 3, 5, 10
	weights	uniform, distance

After running grid search with 5-fold cross validation. The results are shown in Table 4.

Table 4: *Results on Hyper-parameters tuned models*

Model Tuned	Training Accuracy	CV Accuracy
SVM	76.22%	60.02%
Random Forest	75.53%	62.66%
KNN	69.91%	60.09%

SVM and Random Forest perform the best on training data, with minimal differences between them, while KNN shows similar cross-validation accuracy but performs poorly on the training set. In this case, SVM and Random Forest demonstrate good performance in terms of both training accuracy and cross-validation accuracy.

To optimise performance, the best hyper-parameters of SVM and Random Forest are combined using a Stacking Classifier, with Logistic Regression as the meta-estimator. After running on the training dataset, the training accuracy achieved is 78.66%. Since only two models are combined using the Stacking Classifier, this approach does not exceed the allowed model size.

4. Results and Analysis

After running the final model on the test dataset, the best accuracy achieved is 67.7%. Compared to the baseline model accuracy of 56.3%, this model outperforms the baseline by 11.4%. The improvement is attributed to hyper-parameter tuning. Feature engineering and preprocessing also play an important role, as the final model is trained on the image relationship, which provide more meaningful information for the model to train on. The test data undergoes the same feature engineering and preprocessing, which contributes to better results.

This model has been run multiple times on the test data, with results ranging between 65% and 68%. This differences of the results is due to the randomness of the augmentation applied to the training dataset, which effects the hyper-parameter tuning during grid search.

5. Discussion and Conclusions

The face verification model performs impressively well compared to the baseline. This model is trained on very small images, so the information available for the model to learn is limited. Hyper-parameter tuning is highly sensitive for Random Forest. For example, increasing the maximum depth by 5 or reducing the minimum sample split and minimum sample leaf to 2 causes the model to become highly overfitted. Overall, the model benefits from image augmentation and feature engineering.

To further improve the model, fine-tuning the MLPClassifier could result in significant improvements due to the neural network's capability for image detection. Using higher-resolution images in both the training and testing sets could also enhance the model's performance.

6. References

[1] C. M. Bishop, *Pattern recognition and machine learning*, ser. Information science and statistics. New York: Springer, 2006.

[2] A. Géron, *Hands-on machine learning with Scikit-Learn and TensorFlow [electronic resource] : concepts, tools, and techniques to build intelligent systems*. Sebastopol, California: O'Reilly Media, Inc., 2017.

[1] [2]