SYSTÈMES NUMÉRIQUES
POUR L'HUMAIN
ÉCOLE UNIVERSITAIRE DE RECHERCHE

EUR DS4H - Master 1 Informatique
AI models and applications

VGG-Face pretrained embedder and cosine similarity classifiers for celebrity face and sex recognition

Bierhoff Theolien, Hugo Viana, Bastian Holzer

Abstract

This project presents a face recognition pipeline using a pretrained VGG-M Face embedding network with cosine-similarity-based classifiers. Faces are first detected and cropped using a YOLO-based detector, then refined with RetinaFace and Dlib, and mapped to 4096-dimensional L2-normalized embeddings.

The embedding network is frozen during training, while lightweight linear classifiers are optimized. Two configurations are explored: a single-head classifier for celebrity identity, and a multi-head classifier for joint identity and biological sex prediction. Hyperparameters are tuned with Optuna and early pruning.

Evaluation on 105 celebrity identities shows that the single-head model achieves 0.844 accuracy and macro-averaged precision, recall, and F1-score of 0.751, 0.748, and 0.744. The multi-head model reaches 0.728 accuracy for identity (macro-precision/recall/F1: 0.736/0.742/0.744) and 0.938 accuracy for sex prediction (macro-precision/recall/F1: 0.941 /0.940/0.936).

2026

1 Introduction

Face recognition systems generally rely on a multi-stage pipeline including face detection, face alignment or extraction, feature representation, and identity classification. The performance of the overall system depends on the robustness of each of these stages.

In this project, we implement a complete face recognition pipeline using the models assigned in the subject. Face detection and localization are performed using two complementary approaches. First, YOLO26 is used to extract bounding boxes from images in order to localize faces and evaluate detection speed and robustness. In addition, RetinaFace is employed as a dedicated face detection model to produce more precise and reliable face crops, particularly in challenging conditions such as variations in pose or illumination.

Once faces are extracted, identity representation is handled by a pretrained VGG-M Face model with batch normalization. This network is used as a fixed face embedding extractor, producing 4096-dimensional feature vectors for each detected face. The embedding network is frozen during training to leverage pretrained facial representations while reducing computational cost and overfitting.

Face recognition is formulated as a multiclass classification task over 105 celebrity identities. A cosine-similarity-based linear classifier is trained on top of the fixed embeddings. This separation between embedding extraction and classification allows the impact of classifier design and hyperparameters to be studied independently of the backbone model.

The objective of this work is to evaluate the effectiveness of this pipeline, analyze detection and recognition performance, and study the influence of hyperparameter optimization on classification results using standard evaluation metrics.

2 Materials and methods

2.1 Dataset

The dataset used for this project is a public domain face recognition dataset found on Kaggle[1]. It is composed of photos of cropped faces of 105 celebrities. It contains a total of 17,534 photos.

2.2 Models

2.2.1 Bounding boxes extraction

Bounding boxes extraction was carried out using the YOLO26[2] object detection model. This model provides bounding box predictions along with class labels and confidence scores. The model was applied independently to each image in the dataset.

The dataset is organized into subdirectories, each corresponding to a single identity (celebrity). For each subdirectory, all images were processed sequentially. Each image was passed through the YOLO detector, and the inference time was recorded to evaluate the computational cost of the detection step.

[1] https://www.kaggle.com/datasets/hereisburak/pins-face-recognition
[2] https://docs.ultralytics.com/fr/models/yolo26/

For every detected object, the model outputs a bounding box defined by its top-left and bottom-right coordinates, a predicted class label, and a confidence score. The bounding box coordinates were used to crop the original image, extracting the detected region. Each cropped image was converted to RGB format and saved to a new directory structure that mirrors the original identity-based organization.

In addition to saving the cropped face images, detection metadata was collected for evaluation purposes. For each detected bounding box, the following information was stored:

- the celebrity associated with the image,
- the original image name,
- the predicted class label,
- the confidence score of the detection,
- the inference time for the image.

These results were aggregated into a structured table and exported as a CSV file. This file was later used to analyze detection performance and processing efficiency.

2.2.2 Face extraction

To extract faces and their associated geometric information, two models were used: *RetinaFace* and the *Dlib face detector*. This dual approach was adopted to improve robustness and accuracy. First, Dlib is used to detect and roughly localize the face, allowing an initial cropping (masking) of the image. This preprocessing step reduces background noise and improves the subsequent performance of RetinaFace. RetinaFace is then applied to the cropped image to extract precise facial landmarks, which are used to align and center the face.

During this phase, several sanity checks are performed to ensure that poorly oriented or low-quality faces are excluded from the training pipeline, as such samples could negatively impact the final model performance.

A dedicated subdirectory is created to store the cropped and aligned face images. These processed images are later used directly to train both the single-head and multi-head classifiers.

To evaluate this face extraction phase, metadata are collected for each processed image, including:

- the number of images that successfully pass the sanity checks,
- the face detection and processing time per image.

These metrics are aggregated into a structured table and exported as a CSV file. This dataset is subsequently used to analyze detection performance as well as processing efficiency.

2.2.3 Single-head classifier

The face recognition system is composed of two main components: a face embedding network and a classification head. This modular design separates feature extraction from identity classification and allows the use of a pretrained backbone while training only a lightweight classifier.

The face embedding network is based on a pretrained VGG-M Face[3] architecture with batch normalization.

The original classification layer of the backbone is removed and replaced by an identity mapping, allowing the network to output a 4096-dimensional feature vector for each input face image. The resulting feature vectors are then L2-normalized, ensuring that all embeddings lie on a unit hypersphere. This normalization is commonly used in face recognition systems as it stabilizes training and improves the discriminative power of the learned representations.

During training, the embedding network is entirely frozen: its parameters are not updated by backpropagation. This significantly reduces computational cost and limits overfitting, while still benefiting from the strong representational capacity of the pretrained model.

The classification head consists of a single linear layer that maps the 4096-dimensional normalized embeddings to the number of target identities (105 classes in our case). No bias term is used in this layer.

Both the input embeddings and the classifier weights are L2-normalized before computing the linear transformation. As a result, the logits correspond to cosine similarities between embeddings and class weight vectors. These similarities are multiplied by a scaling factor s, which controls the magnitude of the logits before the softmax operation.

This cosine-based formulation, inspired by methods such as CosFace and ArcFace, improves optimization by preventing the logits from becoming too small after normalization and leads to faster convergence and more stable training.

The model is trained using the cross-entropy loss computed from the scaled cosine logits. Only the parameters of the classification head are optimized during training.

Optimization is performed using the AdamW optimizer, which combines adaptive learning rates with decoupled weight decay. Weight decay is applied as a regularization mechanism to reduce overfitting of the classifier.

During training, input face images are first passed through the frozen embedding network to produce normalized embeddings. These embeddings are then classified by the trainable classification head.

Model performance is monitored using accuracy on a validation set. The model achieving the highest validation accuracy is selected and used for final evaluation on the test set. The modular structure of the system allows the trained classifier head to be reused with the same embedding network for inference or future experiments.

2.2.4 Multi-head classifier

The multi-head face recognition system extends the single-head architecture by jointly predicting multiple attributes from a shared facial representation. As in the single-head setting, the system is composed of a frozen face embedding network and a lightweight

[3]https://www.robots.ox.ac.uk/ albanie/pytorch-models.html

trainable classification module. However, instead of a single classifier, multiple heads are used to perform different prediction tasks in parallel.

The face embedding network is identical to the one used in the single-head classifier. It is based on a pretrained VGG-M Face architecture with batch normalization, from which the original classification layer is removed and replaced by an identity mapping. This modification allows the network to output a 4096-dimensional feature vector for each input face image. The resulting embeddings are L2-normalized, ensuring that they lie on a unit hypersphere.

As in the single-head setup, the embedding network is entirely frozen during training. Freezing the backbone significantly reduces computational cost and prevents overfitting, while still leveraging the strong representational power of the pretrained model.

The classification module consists of two independent linear heads that operate on the same normalized embeddings:

- a celebrity identity head, which predicts one of the 105 target identities,
- a sex classification head, which predicts the biological sex of the individual.

Each head is implemented as a linear layer without bias, mapping the 4096-dimensional embedding space to its respective number of output classes. Before computing the linear transformation, both the input embeddings and the classifier weights are L2-normalized. Consequently, the logits produced by each head correspond to cosine similarities between the embeddings and the class weight vectors.

To improve numerical stability and optimization, the cosine similarities of each head are multiplied by a task-specific scaling factor s. Separate scaling factors are used for the identity and sex classification heads, allowing each task to operate at an appropriate logit magnitude. This cosine-based formulation is inspired by metric-learning approaches such as CosFace and ArcFace and encourages better inter-class separation.

Training is performed using a weighted sum of cross-entropy losses, one for each head. Only the parameters of the classification heads are optimized during training, while the embedding network remains frozen. The total loss is defined as:

$$\mathcal{L} = \lambda_{\text{name}}\mathcal{L}_{\text{name}} + \lambda_{\text{sex}}\mathcal{L}_{\text{sex}},$$

where $\mathcal{L}_{\text{name}}$ and $\mathcal{L}_{\text{sex}}$ denote the cross-entropy losses for identity and sex prediction respectively, and λ controls the relative importance of each task.

Optimization is carried out using the AdamW optimizer, which combines adaptive learning rates with decoupled weight decay. Weight decay is applied to the classifier parameters as a regularization mechanism to reduce overfitting.

During training, each input face image is passed through the frozen embedding network to produce a normalized embedding. This embedding is then simultaneously processed by both classification heads to generate identity and sex predictions.

Model performance is monitored using validation accuracy for both tasks. The model achieving the best validation performance is selected for final evaluation on the test set. This multi-head design enables efficient joint learning of identity and sex information while maintaining a compact and modular architecture that can be easily extended to additional facial attributes in future work.

2.3 Evaluation

2.3.1 Metrics

We used a standard accuracy metric to evaluate the three stages of the project:

$$Accuracy = \frac{correct\ classifications}{all\ classifications}$$

In addition to accuracy, we report precision, recall, and F1-score to provide a more detailed evaluation of the classification performance. Since the task is formulated as a multiclass classification problem, these metrics are computed using a macro-averaging strategy, which assigns equal importance to each class regardless of its number of samples.

For a given class k, precision and recall are defined as:

$$Precision_k = \frac{TP_k}{TP_k + FP_k}$$

$$Recall_k = \frac{TP_k}{TP_k + FN_k}$$

where TP_k, FP_k, and FN_k respectively denote the number of true positives, false positives, and false negatives for class k.

The macro-averaged precision and recall are obtained by averaging these quantities over all K classes:

$$Precision_{macro} = \frac{1}{K} \sum_{k=1}^{K} Precision_k$$

$$Recall_{macro} = \frac{1}{K} \sum_{k=1}^{K} Recall_k$$

Finally, the F1-score, which balances precision and recall, is defined as the harmonic mean of precision and recall:

$$F1_{macro} = 2 \cdot \frac{Precision_{macro} \cdot Recall_{macro}}{Precision_{macro} + Recall_{macro}}$$

2.3.2 Tests and validation

For the face embedding and classification step, the dataset was split into 3 subsets: a training set, a validation set and a test set. The training set was used during the training phase for the backpropagation and contains 70% of the data. A major risk during a neural network training is overfitting to the training set, in which case, instead of learning to recognize patterns, the model memorizes the training data. This makes the model very accurate on the training data but not with other data.

To avoid overfitting, a validation set containing 10% of the data is used. For every pass through the training set, the model passes through the validation set without backpropagation and computes the loss. Training stops when the validation loss stops decreasing and starts increasing, indicating overfitting in most cases.

In order to make an unbiased evaluation of the model after the training phase, a test set containing 20% of the data is passed through the model without backpropagation. The outputs are used to compute the accuracy previously described.

2.4 Hyperparameters optimization

Hyperparameters can greatly influence the performance of a model. Unfortunately, the optimal hyperparameters cannot be known without testing. For this study, we tested a large range of values for the hyperparameters to optimize (Table 1).

Hyperparameter	Search space
Learning rate	$[3 \times 10^{-4}, 3 \times 10^{-3}]$
Cosine scaling factor	{16, 32, 48, 64}
Weight decay	$[10^{-5}, 5 \times 10^{-4}]$

Table 1: Search space of the hyperparameters.

Hyperparameter optimization was performed using Optuna, an automatic hyperparameter optimization framework. We conducted 30 trials, where each trial sampled a different combination of hyperparameters from the predefined search space.

To reduce computational cost, each trial was trained for a maximum of 20 epochs. Additionally, Optuna's pruning mechanism was employed to terminate poorly performing trials early when their validation accuracy was significantly worse than that of other trials.

At the end of the optimization process, the combination of hyperparameters yielding the best validation accuracy was selected. These optimal hyperparameters were then used to train the final model for a larger number of epochs.

3 Results

3.1 Bounding boxes extraction

Table 2 reports the image-level reliability of the YOLO-based object detection model used to identify humans within the dataset. All images are known *a priori* to contain at least one human subject, which allows the evaluation to focus on detection reliability rather than localization accuracy.

An image is considered correctly detected if the model predicts at least one bounding box classified as `person`. Under this criterion, the detector achieves a high reliability, indicating that the vast majority of images contain at least one valid person detection. This confirms that the model is well suited as a preprocessing step for downstream face extraction and recognition tasks, where missing a subject would directly propagate errors to later stages.

Table 3 summarizes the inference efficiency and confidence statistics of the YOLO detector. Prediction times are computed on a per-image basis, while confidence statistics are aggregated over all bounding boxes classified as `person`.

The relatively low mean prediction time demonstrates that the detector can be applied efficiently at scale, while the confidence statistics indicate stable and consistent person detections across the dataset. These properties are essential for ensuring both computational feasibility and robustness in the overall face recognition pipeline.

Model	Metric	Value
YOLO (Person)	Total images	17227
	Images with person detected	17058
	Reliability	99.02%

Table 2: Image-level person detection reliability of the YOLO model.

Figure 1 shows the distribution of confidence scores for detections classified as `person`. The distribution is strongly skewed toward high confidence values, with most detections exceeding 0.8. This indicates that the YOLO model not only detects persons reliably but does so with high certainty in the majority of cases. The presence of lower-confidence detections likely corresponds to challenging images involving occlusions, unusual poses, or small-scale subjects.

Model	Metric	Value
YOLO (Person)	Prediction time mean (s)	2.074
	Prediction time std (s)	0.128
	Person confidence mean (%)	84.41
	Person confidence std (%)	16.58

Table 3: Prediction time and confidence statistics for YOLO-based person detection.

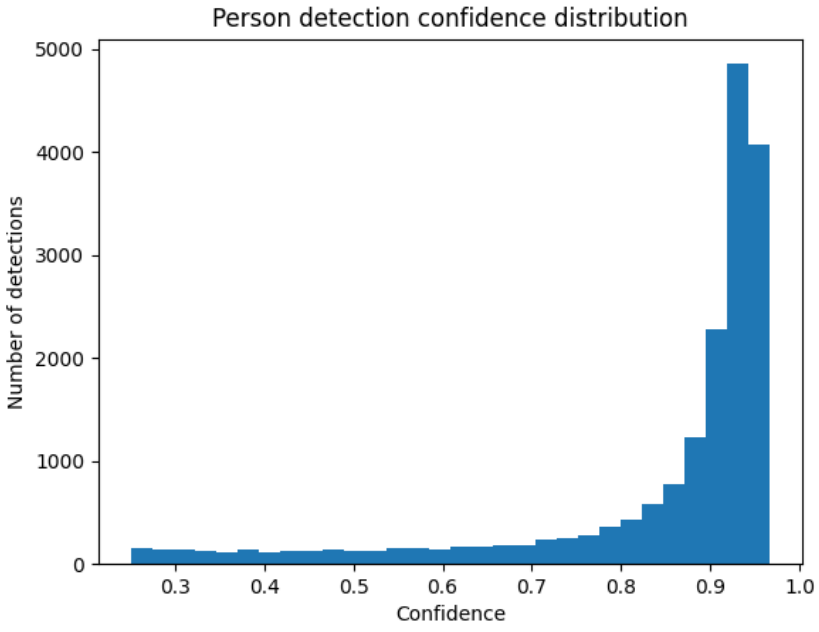

Figure 1: Distribution of confidence scores for detections classified as `person` by the YOLO model.

3.2 Face extraction

Table 4 reports the image-level reliability of the RetinaFace-based face extraction pipeline. Each image was processed by RetinaFace and Dlib to detect, crop, align and center a human face.

An image is considered successfully processed if RetinaFace extracts at least one face. Under this criterion, the pipeline achieves a moderate reliability of 39.6%, reflecting that some images are challenging due to occlusions, extreme poses, or other visual difficulties. Despite the lower extraction rate compared to YOLO detection, the retained faces are expected to be of high quality for subsequent embedding and classification.

Model	Metric	Value
RetinaFace (Face)	Total images	17058
	Images with face extracted	6760
	Reliability	39.63%

Table 4: Image-level face extraction reliability of the RetinaFace model.

Table 5 summarizes the inference efficiency of the RetinaFace pipeline. Prediction times are measured per image.

The mean prediction time of 4.65 seconds per image indicates that while the face extraction step is slower than YOLO-based person detection, it provides high-quality aligned faces necessary for reliable feature embedding and classification.

Model	Metric	Value
RetinaFace (Face)	Prediction time mean (s)	4.65
	Prediction time std (s)	1.12

Table 5: Timing statistics for RetinaFace-based face extraction.

3.3 Face embedding and classification

Table 6 presents the test performances of the face recognition model composed of a fixed face embedding network followed by either a single-head or multi-head linear classifier.

The single-head classifier, trained only on identity labels, achieves an accuracy of 0.844. However, the lower macro-averaged precision, recall, and F1-score (~ 0.75) indicate uneven performance across identities. This suggests that while the model performs well overall, some identities remain more difficult to classify, this can be attributed to the uneven sample size between celebrities.

The multi-head classifier introduces an auxiliary prediction task on sex in addition to identity classification. While identity accuracy decreases to 0.728 in this setting, the auxiliary sex classification achieves a high accuracy of 0.941. This shows that the shared embeddings capture strong demographic information, but that jointly optimizing multiple objectives may negatively impact identity discrimination in this configuration.

Model	Classification feature	Metrics			
		Accuracy	Precision	Recall	F1-Score
Single-head	Identity	0.844	0.751	0.748	0.744
Multi-head	Identity	0.728	0.736	0.742	0.733
	Sex	0.941	0.940	0.936	0.938

Table 6: Test performances of the classifiers.

4 Discussion

4.1 Models performance

As we have seen, the single-head classifier performs relatively better than the multi-head classifier, on predicting the identity of a celebrity. This may be due to the interference of the sex classifier head.

Furthermore, the heterogenous distribution of the dataset between celebrities caused the precision, recall and F1-Score to be significantly lower than accuracy. Unfortunately, for some celebrities, we didn't have enough pictures to make a precise prediction.

4.2 Future works

With the given time for the project, we were unable to broaden our experiments. It would have been interesting to test the performances of a single-head sex classifier, given the encouraging performances with the multi-head.

Additionally, given a larger data sample, fine-tuning the VGGFace model to fit our celebrities could improve the performances.

Code availability

All the code is available at: `https://github.com/realhugoviana/AI-Models_2025`.