FACIAL RECOGNITION-BASED APPROACH FOR EFFICIENT PERSONAL INFORMATION RETRIEVAL IN CYBERSPACE

Hai-Hong Phan[1,*]

Abstract

The paper proposes an artificial intelligence system that can perform face searches on the Internet. Face search is the search for images or videos that contain a specific person's face in cyberspace. The proposed system integrates four key components: the Data Collection Module, Data Storage Mechanism, Search Module, and Analysis Module. By integrating advanced deep learning models, the system achieves high precision and reliability in detecting and recognizing faces. Moreover, our system leverages advanced processing technologies such as parallel computing and optimization techniques to speed up the face search process and handle large-scale data. Experimental results demonstrate that the system achieves remarkable accuracy and efficiency in operation.

Index terms

Face detection; facial recognition; face searching; cyberspace.

1. Introduction

Nowadays, cyberspace has evolved into a valuable and expanding storehouse of information that can be accessed and shared easily from any location. We access cyberspace for various purposes, including communication, education, entertainment, and business. However, along with these benefits come challenges and risks related to security and information handling. Protecting personal information and privacy from malicious actors is an essential requirement. Therefore, promoting research and implementation of artificial intelligence (AI) systems that can accurately identify, track and manage online individual information while maintaining standards of security and reliability is becoming increasingly important.

For the purpose of protecting users' information, we would like to build a system that can assist users in checking whether their personal images are being used illegally

[1]Institute of Information and Communication Technology, Le Quy Don Technical University
[*]Corresponding author, email: hongpth@lqdtu.edu.vn DOI: 10.56651/lqdtu.jst.v13.n02.922.ict

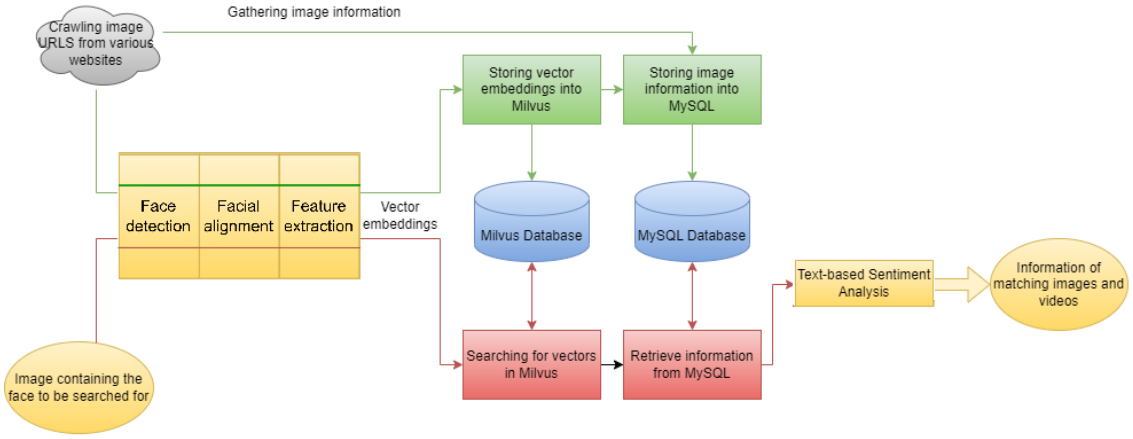

Fig. 1. The pipeline of the proposed method is illustrated as follows: The Data Collection Module is depicted in grey, the Data Storage Module in green, the Search Module in red, and the Analysis Module in yellow.

online. To do this, we leverage face recognition technology as a basis to build the application. By taking advantage of each individual's unique facial features, facial recognition technology [1]–[3] can identify, track and manage user information in cyberspace. Implementing it offers many benefits, including convenience, accuracy, and personalization. Users can automatically control whether their personal information is posted in certain places on the internet. From there, they know how their personal images are used online, and the system will notify users of cases of unauthorized use of their images.

Deep learning techniques have recently paved the way for facial recognition methods. Notable face recognition models like FaceNet [4], ArcFace [5], and YOLOFace [2] have achieved outstanding results. These models can achieve accuracy rates of over 99% in controlled environments. Techniques like transfer learning, data augmentation, and architectural innovations have steadily improved the ability to match faces accurately.

However, integrating facial recognition into applications in cyberspace with fast processing times is a real challenge. In cyberspace, facial recognition models often need to communicate with remote servers or the cloud to process the data. Network latency can introduce significant delays, hampering the ability to achieve real-time processing speeds. Running advanced facial recognition models requires significant computational power, which may not always be available, especially on mobile or edge devices with limited resources. Scaling facial recognition to handle large volumes of users or requests in cyberspace can be difficult, as the system needs to maintain low latency and high throughput despite increasing workloads. To solve the above issues, we introduce an innovative system called MTAFaceSearchV2, designed to utilize facial recognition technology to precisely, and quickly retrieve personal information from cyberspace. Our system comprises four primary components: the Data Collection Module, the Data Storage Mechanism, the Search Module and

Analysis Module, as illustrated in Fig. 1. Together, these components work in parallel to ensure accuracy, robustness, and efficiency in the retrieval process.

- The responsibility of the Data Collection Module is to acquire faces from images and videos from a list of internet links. This list can be updated and changed according to the user's intentions at different times. This module employs advanced deep learning models to extract facial features, then stored temporarily in the database. This process is depicted in the green branch.
- The Data Storage Mechanism used in our system exploits the power of Milvus and MySQL databases. This combination enables us to efficiently store and retrieve face-related information, providing a scalable solution for large-scale face recognition applications. The mechanism also facilitates fast and accurate face searches.
- The Search Module works by receiving an image containing the registered user's own face and matching it with the image obtained from the Data Collection Module. For images found that belong to the user, this module will retrieve information related to the image including links and text information if any. This process is shown in the red branch.
- The Analysis Module plays an important role in face recognition and conducting sentiment analysis of text containing searched faces. This process is depicted in the yellow branch.

Our system provides users with a tool to manage their image and reputation in cyberspace effectively. Users are only allowed to check their images and are not allowed to search for images related to anyone else to ensure each person's privacy.

2. Related works

In recent years, various face search methods leveraging AI have gained significant attention, particularly in areas like security, surveillance, and identity management. Systems such as Clearview AI [6] and Amazon Rekognition [7] have demonstrated the ability to search for faces across massive image datasets, using deep learning to achieve high accuracy in identifying individuals even from low-quality images. Clearview AI [6], for instance, is employed by law enforcement agencies to match faces against publicly available photos from social media and websites. Similarly, Face++ [8], developed by Megvii, excels in real-time facial recognition and is used extensively in public security systems in China. Microsoft Azure's Face API [9] and Trueface [10] focus on providing secure access and identity verification through cloud-based face matching, offering robust solutions for corporate security and authentication. Another notable method is NTechLab's FindFace [11], used in large-scale surveillance systems, which achieves rapid and accurate face matching in crowded environments. These systems have revolutionized applications from airport security and border control to retail analytics, but they also raise concerns regarding privacy and data protection. This body of work highlights the rapid evolution of face search technology and its diverse applications across industries.

The facial recognition process involves the extraction of distinct features from a person's face, including facial shape, the positioning of facial landmarks, and other distinguishing characteristics. These extracted features are subsequently compared to a pre-existing database of known faces. In the event that the extracted features align with a face within the database, the individual is identified as the person associated with that particular face.

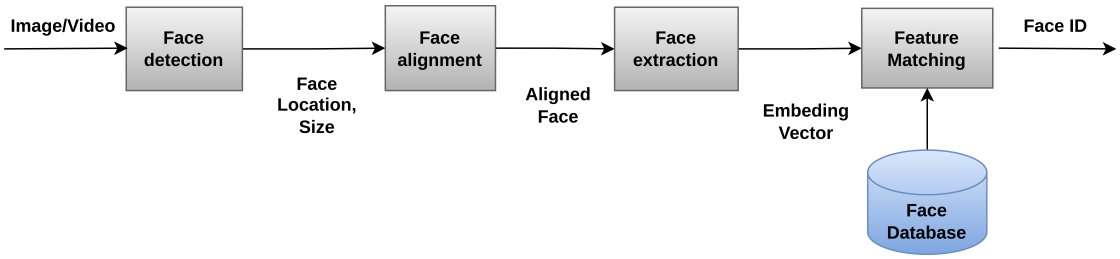

Fig. 2. General model for face recognition methods.

Facial recognition encompasses multiple approaches. Modern facial recognition systems commonly comprise four key components, each corresponding to a distinct step: face detection, face preprocessing (alignment, normalization), feature extraction, and face classification. These sequential steps collectively contribute to the overall process of facial recognition, as illustrated in Fig. 2.

The initial stage of facial recognition, known as face detection, aims to locate and determine the size of faces within an image or video. This stage has witnessed advancements through various classical and modern methods. Classical methods, such as the Haar cascades introduced by Viola and Jones, employ edge analysis and rectangular features for efficient face detection [1]. Another classical approach is template matching, which involves comparing an image with templates to identify similarities, although it can be influenced by scale, rotation, and lighting conditions.

Recent progress in deep learning has propelled the development of powerful face detection techniques. Convolutional Neural Networks (CNNs) have revolutionized face detection by enabling automatic feature extraction from images. MTCNN (Multitask Cascaded Convolutional Networks) employs a cascaded CNN approach to detect faces on different scales and refine the results [12]. RetinaNet uses a single network to propose regions of interest and predict their classification and localization simultaneously, enhancing face detection in varying lighting conditions [13]. Additionally, YOLOFace [2], an extension of the popular YOLO architecture [14]–[16], specializes in efficient face detection by optimizing for various facial regions and scenarios. These modern techniques highlight the significance of deep learning in achieving state-of-the-art performance in face detection.

The next step, Feature Extraction from Faces, focuses on extracting crucial features from the preprocessed facial data. These features encompass key facial points such as the positions of the eyes, nose, and mouth, as well as geometric characteristics like

distribution vectors of facial landmarks. Various methods can be employed to extract facial features, including PCA [17], LDA [18], and deep learning models such as CNN [19], [20], FaceNet [4], and ArcFace [5]. These techniques utilize hypersphere embedding to learn discriminative features specifically for face recognition. By effectively handling considerable intra-class variations and minimizing inter-class distances, they enhance the accuracy and precision of face recognition tasks.

Once the features of the face images are extracted, a similarity measurement algorithm is applied to assess the resemblance between them. Common similarity metrics include Euclidean distance, cosine similarity, or correlation-based measures. These metrics quantify the dissimilarity or similarity between the features, representing how closely the faces match. During face matching, the similarity score obtained from the similarity measurement indicates the degree of likeness between two faces. If the similarity score surpasses a predefined threshold, the faces are considered a match, suggesting that they belong to the same individual. Otherwise, they are deemed different individuals.

3. Proposed method

We propose the face search system MTAFaceSearchV2 applying facial recognition technology, consists of four main components: the Data Collection Module, Data Storage Mechanism, the Search Module and the Analysis Module. The Data Collection Module is responsible for gathering faces from images and videos available on the internet. It extracts facial features and stores them in the database. The Data Storage Mechanism leverages the capabilities of Milvus [21] and MySQL databases. Meanwhile, the Search Module takes as input an image containing the face of the person to be searched for. It extracts feature vectors and searches for them in the collected database, returning the images and links to web pages that contain relevant information about the person. The Analysis Module assumes a critical role in face recognition and incorporating sentiment analysis of text that contains the searched faces. This module allows the system to analyze the emotions expressed in text containing faces, thereby determining the overall intent of that text. We integrate state-of-the-art deep learning models that can detect and recognize faces with high accuracy, robustness and speed.

3.1. Data collection module

Collecting images and content manually from internet can be very timeconsuming, especially when dealing with vast quantities. Leveraging automation tools, such as BeautifulSoup [22], Scrapy [23] and Selenium [24] in Python, dramatically accelerate this process. In our workflow, we harnessed the power of these tools and libraries to systematically generate code that downloads images from the web. The data collection process from the internet can involve various sources such as websites, Google Images, Facebook, and YouTube. To accomplish this, libraries like `requests`, `urllib`, `pafy` (based on `youtube-dl`), and `facebook-scraper` are utilized to scan, gather, and extract images and videos.

The process of collecting images from websites involves the following steps:
Step 1. Use the requests library to fetch the content of the website and convert it into HTML format.
Step 2. Use the BeautifulSoup library to parse the website's content and extract image URLs found on the website.
Step 3. Use the requests library again to fetch the images from their respective URLs.
Step 4. Eliminate images that are logos, icons, etc., retaining only the final desired images.
To enhance the efficiency of data collection, we can implement automated scanning of URLs within a domain, as illustrated in the Fig. 3.

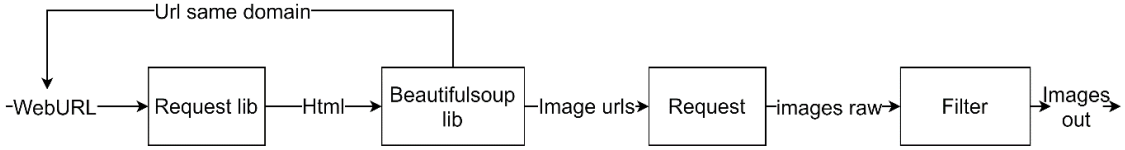

Fig. 3. Illustration of automated scanning of URLs within a domain.

YouTube is a social media platform and online video-sharing network where users can upload or view videos on various devices. The libraries used to collect videos from YouTube are scrapetube and pafy. Firstly, the scrapetube library is used to obtain video IDs from a keyword, a YouTube channel, or a playlist. Afterwards, the pafy library is employed to convert these video IDs into URLs that can be directly read by OpenCV.

Selenium is a versatile and powerful tool for automating web tasks to collect data from Facebook, making it a valuable asset for web scraping and data extraction. However, recent changes in Facebook's policies have posed challenges for data scraping from the platform. Due to evolving privacy policies, we encountered increased difficulty in retrieving data from Facebook. To adapt to Facebook's evolving security measures, we employed 2FA in conjunction with Selenium. This adjustment allowed us to circumvent potential issues associated with data scraping and account suspension.

The extracted images and videos are then passed through the face detection module (i.e. YuNet [3]) to locate the positions of faces in the images and videos. For videos, MIL tracker [25] is used for tracking to enhance processing speed. YuNet [3] not only determines the position and size of the faces but also provides five key facial landmarks.

Based on these five key facial landmarks, the faces are aligned and cropped from the original images before passing through the feature extraction module (i.e. ArcFace [5]). A 512-dimensional vector is generated for each face, and these vectors are stored in the Milvus database for fast retrieval. Each vector is assigned an index, which serves as a key to link the stored vectors in the Milvus database with information about the origin of the stored images and videos. Fig. 4 shows the connection of the Data Collection Module to the Data Storage Mechanism Module and the Analysis Module. A notable feature of this module is that the system does not store the facial images directly; instead, they are stored temporarily as feature vectors. This approach addresses security concerns and

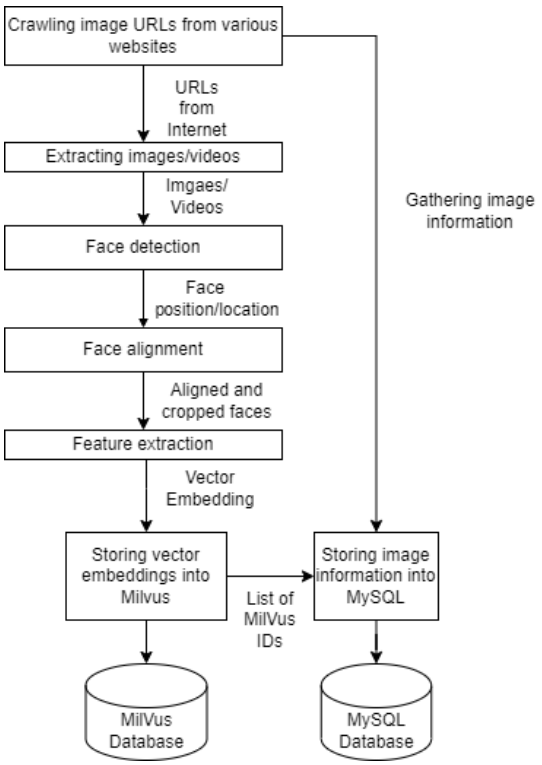

Fig. 4. Illustration of the proposed Data Collection Module and its connection to Data Storage Mechanism Module and Analysis Module.

ensures compliance with the intellectual property rights of the images on the internet.

3.2. Data storage mechanism module

One of the main challenges in face recognition systems, especially for large-scale applications, is the efficient storage and retrieval of facial feature vectors. These vectors contain essential information about each face and are used for recognition and matching algorithms. The system's capability is critical when dealing with large amounts of data, and the performance depends on the effectiveness of data organization and search methods.

We propose a data storage and retrieval model that uses Milvus and MySQL databases (as illustrated in Fig. 5). Milvus provides efficient indexing and querying of high-dimensional vectors, which is suitable for storing facial feature vectors. MySQL offers robust data management and relational capabilities, which allows us to store additional information related to the faces, such as URLs and face locations.

When the system crawls data, with the faces collected in the list of URLs, the information will be stored in the SQL database and the Milvus database specifically as follows: In SQL, each URL will store a list of images it contains, with each image will store a list of faces on the image. For each face on the images, the embedding

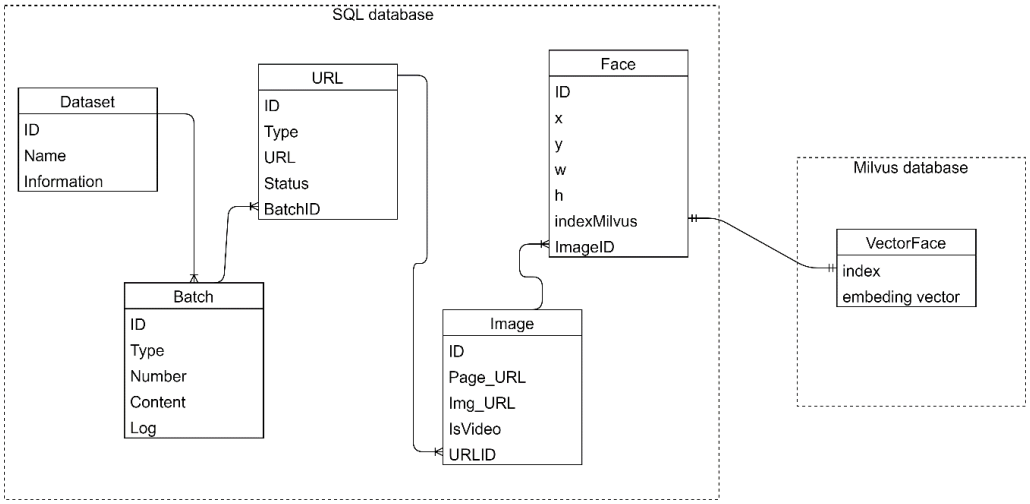

Fig. 5. Illustration of the proposed Data Storage Mechanism.

vector will be extracted to save in the Milvus database. Milvus only saves the embedding vector of the face with the index generated in that database. Therefore, when there is an input face that needs to be searched for whether it appears in the list of the URLs, the face will be extracted for embedding vector and compared with the embedding vectors in the Milvus database to find out the most matching vectors. From there, the system links to the SQL database to find out the faces that match the input image, the images and the image links containing these faces. The result returns a list of these links and the images containing the faces that need to be searched.

The module allows us to store and retrieve face-related information effectively, providing a scalable solution for large-scale face recognition applications. On the other hand, the separation of facial feature vectors and associated information into two databases (i.e., Milvus and MySQL) allows us to optimize the performance and manage the data efficiently. This approach is crucial for real-time applications, where quick and accurate face recognition results are essential.

3.3. Search module

The input of the module is an image containing the face to be searched for. When the image contains multiple faces, the system provides an interface for users to choose the specific face they want to search for. After going through the face detection, alignment, and feature extraction steps as in the data collection module, the resulting feature vectors will be searched in the vector database.

By using Milvus, we can efficiently perform similarity queries on the facial feature vectors, thus identifying faces with high degrees of similarity to the input query vector. This capability is essential in face recognition applications, where we aim to identify known faces or detect potential matches among a large database of faces.

The similarity search is usually performed using distance metrics such as Euclidean

distance or inner product, depending on the requirements of the face recognition system. These distance metrics help measure the similarity between vectors and help select the most relevant results during the search process. By using a predetermined threshold, we can control the quality of the search results and ensure that only highly similar facial feature vectors are retrieved.

After we have obtained the indices of the facial feature vectors with high similarity, we retrieve additional information related to these faces from the MySQL database. This information may include URLs to the original images or face locations in the images. By combining the power of Milvus for similarity searches and the data management capabilities of MySQL, we create a robust and efficient system for face recognition.

The output at this step will be a list of indices corresponding to similar faces to the one being searched for. The information about matched faces will be queried based on the list of indices and returned to the user. In the search module (as illustrated in Fig. 1), we prioritize the system's accuracy over speed compared to the data collection module.

3.4. Analysis module

The Analysis Module plays an important role in face recognition and conducting sentiment analysis of text containing searched faces. The Analysis Module helps detect and recognize faces as input for the Data Storage Mechanism and Search Module. At a later stage, Sentiment Analysis receives information from the Search module to analyze text sentiment, providing users with more information about the posting source.

3.4.1. Face detection with YuNet model: In this research, we integrate YuNet [3] as the face detection module. YuNet model was introduced by Wei Wu and colleagues in 2021, optimized for a balance between speed, resources, and accuracy, making it suitable for tasks that require real-time processing.

The key differentiation between YuNet and other models is its incorporation of a compact and efficient feature extraction backbone, alongside a simplified pyramid feature fusion neck (Fig. 6). In addition, a training strategy for the compact face detector is outlined, enabling effective model training while maintaining the same data distribution as the training set. YuNet achieves the optimal balance between accuracy and speed. The model excels in accurately detecting faces within images and videos, even when faces are obscured or distorted. Notably, YuNet's standout feature is its ultralightweight design, allowing for easy deployment on devices without requiring high-end hardware specifications.

After identifying the face's position, the algorithm proceeds to perform face alignment and normalization. It includes resizing the face to ensure consistent input sizes for subsequent steps, rotating, cropping, or transforming the face to a standard form. In this paper, we perform face alignment based on five key facial landmarks for pre-processing.

3.4.2. Face recogintion with ArcFace and Milvus: The key innovation of the ArcFace method [5] is the use of an Additive Angular Margin Loss function. This loss function enforces two important properties in the learned embedding space: 1) compactness of

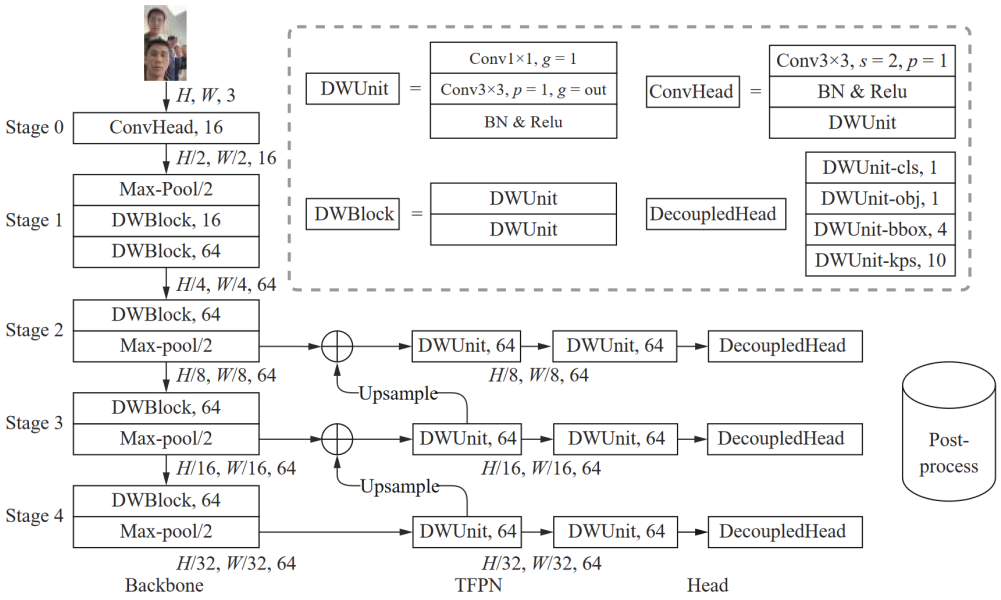

Fig. 6. Illustration of YuNet's architecture [3].

embeddings for samples belonging to the same class (i.e. the same person), and 2) increased discrepancy between embeddings of different classes (i.e. different people). This helps to improve the discriminative power of the learned face representations.

The mathematical formulas of an Additive Angular Margin loss (AAM for abbreviation) are as follows:

$$L_{AAM} = -\log \frac{e^{s\cos(\theta_{y_i}+m)}}{e^{s\cos(\theta_{y_i}+m)} + \sum_{j=1,j\neq y_i}^{N} e^{s\cos(\theta_j)}}$$

ArcFace algorithm is as follows:
Step 1. Normalize vector x and columns of weight matrix W.
Step 2. Compute W_x.
Step 3. Calculate $cos(\theta_j)$ between each column of matrix W and x.
Step 4. Compute $arccos(\theta_j)$ between each column of matrix W and x.
Step 5. Add θ of the ground truth class with margin m (angle in radians).
Step 6. Multiply logit value by x_i.
Step 7. Calculate softmax loss as in the formula above.

We integrate ArcFace model [5] due to its advantages such as high accuracy, robustness, stability and relatively simple training model. In Fig. 7, the left graph shows the image feature without an additive angular margin penalty, while the right graph shows the image feature with it. As you can see, the one with an Additive Angular Margin loss (right) has more dense image feature space in the intra-class but is more discriminative in the inter-class than the one without it (left).

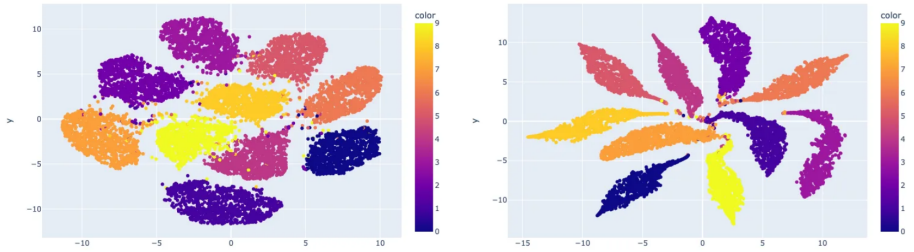

Fig. 7. (Left) the image feature without AAM; (Right) the image feature with AAM [26].

To speed up the classification process, we apply Milvus [21]. Milvus is an open-source vector database system designed to store and query large-scale vector data. It supports building and managing vector spaces, providing efficient search and clustering functions for vector data. Milvus offers high scalability, allowing processing of large datasets and supporting complex vector computations.

3.4.3. Sentiment analysis: The text-based sentiment analysis section will provide metrics regarding the article's sentiment. In the context of this project, we are utilizing a pre-trained PhoBERT model [27] version of "base" fine-tuned on a substantial dataset of 30,000 e-commerce-related texts. This model categorizes the sentiment percentages, encompassing negative, positive, or neutral sentiments, for the sentences associated with the collected images. This comprehensive analysis provides us with a holistic view of the search data.

PhoBERT pre-training approach is based on RoBERTa which optimizes the BERT pre-training procedure for more robust performance. PhoBERT [27] stands as an intricate adaptation of BERT meticulously crafted to cater to the nuances of the Vietnamese language. By offering contextual embeddings for both words and sentences in Vietnamese, PhoBERT [27] has emerged as an invaluable asset for developers and researchers delving into Vietnamese language recognition.

4. Results and evaluation

4.1. Experiment settings

We evaluate the system on two datasets: The proposed face dataset, namely MTAFace, and CelebA dataset [28]. The MTAFace dataset contains total of 5,721 images of 53 individuals, is captured from camera with a resolution of 1,280 x 720 pixels. This dataset has the low-quality images and the various angles of faces. We divide this dataset into 2 parts: 80% for training and 20% for testing. The Fig. 8 shows some of images from the dataset. The CelebA dataset [28] is a large-scale face dataset consisting of 202,599 images, 10,177 identities. It is also divided into 20% for testing and 80% for training.

The system configuration is as follows: CPU - 11th Gen Intel® Core™ i5-11400H@2.70GHz × 12, RAM - 16GB, NVIDIA GeForce RTX 3050.

Fig. 8. Illustration of the proposed MTAFace dataset.

4.2. Face recognition results

In this section, we used the MTAFace dataset to assess the accuracy and processing speed of three methods. FaceNet (128 and 512) [4], ArcFace [5], and SFace [29].

As shown in Table 1, ArcFace achieves remarkably high accuracy on a custom data set consisting of moderate quality real-world camera images and various facial orientations for each individual. Moreover, ArcFace maintains a reasonable processing speed threshold. Based on these results, we have decided to use ArcFace as the face recognition model in our system.

Table 1. Comparison of time and accuracy between Face recognition methods on MTAFace dataset

Model	Embedding Size	Model Size (MB)	Time (s)	Accuracy (%)
SFace	128	38.7	0.01	77.80
FaceNet	128	92.2	0.083	76.25
FaceNet512	512	95	0.08	81.50
ArcFace	512	137	0.109	**86.55**

To evaluate the accuracy of the proposed module (including YuNet, ArcFace and Milvus), we use the CelebA dataset [28]. Fig. 9 shows the results of our evaluation, using the following metrics: Precision - The ratio of correctly identified faces to the total number of detected faces; Recall - The ratio of correctly identified faces to the total number of faces in the CelebA dataset; F1 score - A harmonic mean of Precision and Recall that measures the overall accuracy of the system. We can see that by adjusting the maximum Euclidean distance, we can achieve a high F1 score of 0.71, with a Precision score of 0.85, while keeping Recall at an acceptable level. This indicates that the proposed module is a reliable and robust model for face recognition.

4.3. Effectiveness of Milvus

We compared the performance of the Cdist [30] and Milvus [21] methods using a dataset of facial images for calculating cosine similarity. We searched for 1,000 samples from a total of one million vectors in each measurement, and we reported the results in Table 2. We observed a significant speed difference between Milvus and Cdist with 128-dimensional vectors: Milvus on CPU was about 7 times faster than Cdist on CPU, while Milvus on GPU surpassed Cdist on CPU by a factor of 15. This is because Milvus

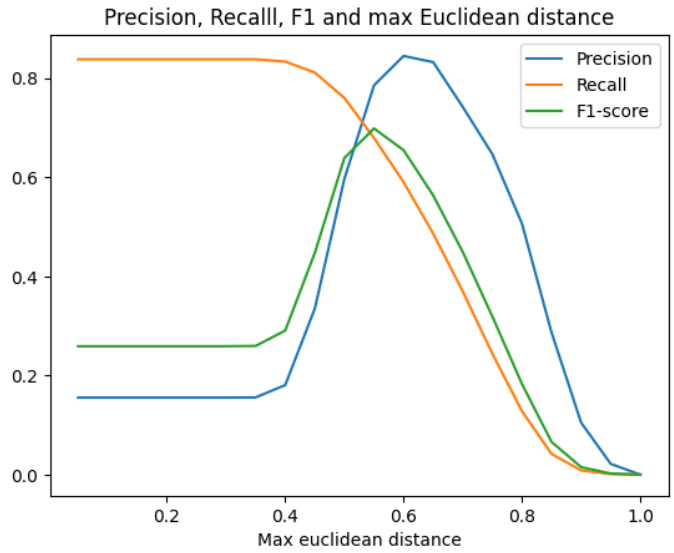

Fig. 9. The influence of Max-Euclidean Distance on the proposed module accuracy (including YuNet, ArcFace and Milvus) on the CelebA dataset.

is optimized for large datasets and uses a more efficient implementation of the cosine similarity algorithm.

Table 2. Comparison of average time (seconds) to search an image of Cdist and Milvus in a set of one million vectors

Method	Dimension vector	Hardware	Average Time (s)
Cdist (Scipy)	128	CPU	0.107
Milvus	128	CPU	0.015
Milvus	128	GPU	0.008
Milvus	512	GPU	0.024

4.4. Performance of the proposed system

After implementing the system, we evaluated the search speed as the number of faces collected gradually increased, as shown in Table 3.

Table 3. The performance of the proposed system on some test cases. Time is averaged per image

Test case	Number of Faces	Detect and Extract time (s)	Milvus Search time (s)	SQL Search time (s)	Total time (s)
1	5,000	0.047	0.059	0.001	0.107
2	50,000	0.059	0.099	0.036	0.194
3	500,000	0.054	0.199	0.343	0.596
4	1,000,000	0.055	0.510	2.097	2.662

From Table 3, it is evident that in datasets with fewer than 500,000 faces, the total processing time remains under one second. When increasing the number of faces to

nearly one million, the total processing time still hovers around just over 2.5 seconds. This represents a relatively fast processing speed, especially when compared to the vast quantity of faces being searched.

Table 3 shows the time of each module and the total processing time of our system for different sizes of facial datasets. We can see that the proposed system achieves good speed, especially considering the large number of faces that are being searched. For datasets with less than 500,000 faces, our system can complete the search in less than 0.6 seconds. For datasets with nearly one million faces, our system can finish the search in about 2.5 seconds. This shows that the system is very feasible in practice.

4.5. Deploying the system

We deploy the system on the WEB platform using the Django framework in Python, namely MTAFaceSearchV2.

Fig. 10. Uploading images and selecting the faces to search in the MTAFaceSearchV2 application.

Our system provides a user-friendly interface for uploading images and selecting the desired face for searching. Fig. 10 illustrates how the user can upload an image from their device or enter a URL of an online image, and then choose the face they want to search for from the detected faces in the image. With a face search case, we found that the search system is very fast, returning search results in only about 2 - 3 seconds. The search results are sorted by facial similarity. Fig. 11 shows how the system displays the information about the found images, such as the image itself, the URL of the web page where it was found, and sentiment analysis of text containing searched faces.

5. Conclusions

In conclusion, an AI-driven system capable of conducting face searches on the internet is presented in the paper. Leveraging the power of face recognition technology, we have

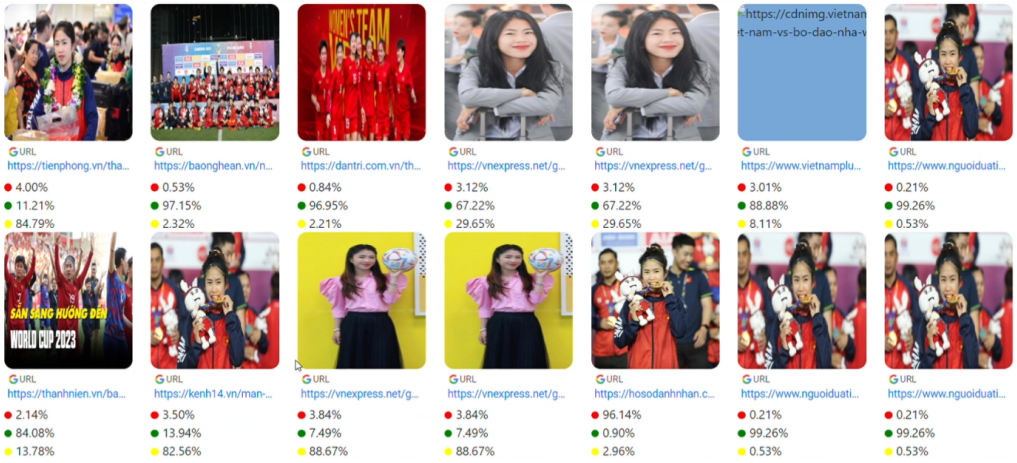

Fig. 11. The interface displays the results of the MTAFaceSearchV2 application: the URL of the web pages and sentiment analysis of text - Red, blue, and yellow correspond to negative, positive and neutral.

successfully developed a system that offers dependable and efficient retrieval of personal information. Our system includes four primary components: the Data Collection Module, the Data Storage Mechanism, the Search Module and the Analysis Module. The system's ability to recognize individuals based on facial features enables detecting fraudulent or impersonation activities and unreliable behaviors like fake news or phishing attempts.

References

[1] L. Cuimei, Q. Zhiliang, J. Nan, and W. Jianhua, "Human face detection algorithm via Haar cascade classifier combined with three additional classifiers," in *13th IEEE International Conference on Electronic Measurement & Instruments (ICEMI), China*, 2017, pp. 483–487. doi: 10.1109/ICEMI.2017.8265863

[2] W. Chen, H. Huang, S. Peng, C. Zhou, and C. Zhang, "YOLO-face: A real-time face detector," *The Visual Computer*, vol. 37, pp. 805–813, 2021. doi: 10.1007/s00371-020-01831-7

[3] W. Wu, H. Peng, and S. Yu, "YuNet: A tiny millisecond-level face detector," *Machine Intelligence Research*, vol. 20, no. 5, pp. 656–665, 2023. doi: 10.1007/s11633-023-1423-y

[4] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, USA*, 2015, pp. 815–823. doi: 10.1109/CVPR.2015.7298682

[5] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "ArcFace: Additive angular margin loss for deep face recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, USA*, 2019, pp. 4690–4699. doi: 10.1109/CVPR.2019.00482

[6] Clearview AI, "The Power of Facial Recognition in U.S. Federal Government," *Clearview AI*, December 30, 2024. [Online]. Available: https://www.clearview.ai/federal, [Accessed: December 30, 2024].

[7] Amazon Web Services, "Amazon Rekognition: AI-based Image and Video Analysis," *Amazon Web Services*, December 30, 2023. [Online]. Available: https://aws.amazon.com/rekognition/, [Accessed: December 30, 2024].

[8] Megvii Technology, "FaceID The world's leading face-based identity verification service," *Megvii*, December 30, 2024. [Online]. Available: https://www.faceplusplus.com/faceid-solution/, [Accessed: December 30, 2024].

[9] Microsoft Corporation, "Face API Pricing," *Microsoft Corporation*, November 30, 2023. [Online]. Available: https://azure.microsoft.com/en-us/pricing/details/cognitive-services/face-api/, [Accessed: December 30, 2024].

[10] Trueface, "The world's leading computer vision solutions," *Trueface*, December 30, 2024. [Online]. Available: https://www.trueface.ai/our-technology, [Accessed: December 30, 2024].

[11] NtechLab, "Findface Multi," *NtechLab*, December 30, 2024. [Online]. Available: https://ntechlab.com/findface-multi/, [Accessed: December 30, 2024].
[12] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499–1503, 2016. doi: 10.1109/LSP.2016.2603342
[13] J. Deng, J. Guo, E. Ververas, I. Kotsia, and S. Zafeiriou, "RetinaFace: Single-shot multi-level face localisation in the wild," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, USA*, 2020, pp. 5203–5212. doi: 10.1109/CVPR42600.2020.00525
[14] Z. Yu, H. Huang, W. Chen, Y. Su, Y. Liu, and X. Wang, "YOLO-Facev2: A scale and occlusion aware face detector," *Pattern Recognition*, vol. 155, 2024. doi: 10.1016/j.patcog.2024.110714
[15] M. Loey, G. Manogaran, M. H. N. Taha, and N. E. M. Khalifa, "Fighting against COVID-19: A novel deep learning model based on YOLO-v2 with resnet-50 for medical face mask detection," *Sustainable Cities and Society*, vol. 65, 2021. doi: 10.1016/j.scs.2020.102600
[16] D. Garg, P. Goel, S. Pandya, A. Ganatra, and K. Kotecha, "A deep learning approach for face detection using YOLO," in *2018 IEEE Punecon, India*, 2018, pp. 1–4. doi: 10.1109/PUNECON.2018.8745376
[17] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 2, no. 4, pp. 433–459, 2010. doi: 10.1002/wics.101
[18] F. Z. Chelali, A. Djeradi, and R. Djeradi, "Linear discriminant analysis for face recognition," in *2009 International Conference on Multimedia Computing and Systems, Morocco*, 2009, pp. 1–10. doi: 10.1109/MMCS.2009.5256630
[19] G. Hu, Y. Yang, D. Yi, J. Kittler, W. Christmas, S. Z. Li, and T. Hospedales, "When face recognition meets with deep learning: An evaluation of convolutional neural networks for face recognition," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops, Chile*, 2015, pp. 142–150. doi: 10.1109/ICCVW.2015.58
[20] R. Chauhan, K. K. Ghanshala, and R. Joshi, "Convolutional neural network (CNN) for image detection and recognition," in *2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC), India*, 2018, pp. 278–282. doi: 10.1109/ICSCCC.2018.8703316
[21] J. Wang, X. Yi, R. Guo, H. Jin, P. Xu, S. Li, X. Wang, X. Guo, C. Li, X. Xu *et al.*, "Milvus: A purpose-built vector data management system," in *Proceedings of the 2021 International Conference on Management of Data, USA*, 2021, pp. 2614–2627. doi: 10.1145/3448016.3457550
[22] Leonard Richardson, "Beautiful Soup," *Crummy*, January 02, 2024. [Online]. Available: https://www.crummy.com/software/BeautifulSoup/, 2004, [Accessed: December 30, 2024].
[23] Selenium, "The Selenium Browser Automation Project," *Selenium*, November 17, 2024. [Online]. Available: https://www.selenium.dev/documentation/, [Accessed: December 30, 2024].
[24] Scrapy, "Scrapy 2.12 documentation," *Scrapy*, November 19, 2024. [Online]. Available: https://docs.scrapy.org/en/latest/, [Accessed: December 30, 2024].
[25] N. Wang, W. Zhou, Q. Tian, R. Hong, M. Wang, and H. Li, "Multi-cue correlation filters for robust visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, USA*, 2018, pp. 4844–4853. doi: 10.1109/CVPR.2018.00509
[26] Yuki Shizuya, "ArcFace Architecture and Practical Example: How to Calculate the Face Similarity Between Images," *Medium*, February 4, 2024. [Online]. Available: https://medium.com/@ichigo.v.gen12/arcface-architecture-and-practical-example-how-to-calculate-the-face-similarity-between-images-183896a35957, [Accessed: December 30, 2024].
[27] D. Q. Nguyen and A. T. Nguyen, "PhoBERT: Pre-trained language models for Vietnamese," in *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2020, pp. 1728–1738. doi: 10.18653/v1/2020.findings-emnlp.92
[28] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of the IEEE International Conference on Computer Vision, Chile*, 2015, pp. 3730–3738. doi: 10.1109/ICCV.2015.425
[29] F. Boutros, M. Huber, P. Siebke, T. Rieber, and N. Damer, "SFace: Privacy-friendly and accurate face recognition using synthetic data," in *2022 IEEE International Joint Conference on Biometrics (IJCB), United Arab Emirates*, 2022, pp. 1–11. doi: 10.1109/IJCB54206.2022.10007961
[30] SciPy community, "SciPy API," *SciPy*, December 30, 2024. [Online]. Available: https://docs.scipy.org/doc/scipy/reference, [Accessed: December 30, 2024].

Manuscript received 15-4-2024; Accepted 27-12-2024.

■

Hai-Hong Phan received her Ph.D. in computer science from the University of CY Cergy Paris, France in 2019. She has over a decade of experience in research, teaching, consulting, and implementing information technology projects. Her research focuses on computer vision, image processing, action recognition, and face recognition. She is currently a lecturer at Le Quy Don Technical University. E-mail: hongpth@lqdtu.edu.vn.

TRUY XUẤT THÔNG TIN CÁ NHÂN HIỆU QUẢ TỪ KHÔNG GIAN MẠNG DỰA TRÊN NHẬN DẠNG KHUÔN MẶT

Phan Hải Hồng

Tóm tắt

Bài báo đề xuất một hệ thống trí tuệ nhân tạo (AI) có thể thực hiện tìm kiếm khuôn mặt trên Internet. Tìm kiếm khuôn mặt là tìm kiếm hình ảnh hoặc video có chứa khuôn mặt của một người cụ thể trong không gian mạng. Hệ thống được đề xuất tích hợp bốn thành phần chính: Mô đun thu thập dữ liệu, Mô đun lưu trữ dữ liệu, Mô đun tìm kiếm dữ liệu và Mô đun phân tích dữ liệu. Bằng cách tích hợp các mô hình học sâu tiên tiến, hệ thống đạt được độ chính xác và độ tin cậy cao trong việc phát hiện và nhận dạng khuôn mặt. Hơn nữa, hệ thống của chúng tôi tận dụng các công nghệ xử lý tiên tiến như tính toán song song và các kỹ thuật tối ưu hóa để tăng tốc quá trình tìm kiếm khuôn mặt và xử lý dữ liệu quy mô lớn. Kết quả thử nghiệm chứng minh rằng hệ thống đạt được độ chính xác và hiệu quả tốt trong quá trình vận hành.

Từ khóa

Phát hiện khuôn mặt; nhận dạng khuôn mặt; tìm kiếm khuôn mặt; không gian mạng.