Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

Rubik's Encryption and CNN Based Biometric Authentication

[1] Mrs. Divya P
Dept of CSE
Assistant Professor
ACS College of Engineering
Bengaluru.
divyapacs@gmail.com

[2] Ms. Lisha V
Dept. of CSE
Student
ACS college of Engineering
Bengaluru
Lishabv2003@gmail.com

[3] Mr. Kiran M
Dept. of CSE
Student
ACS college of Engineering
Bengaluru
kiranmunirathnam11@gmail.com

[4] Mr. Mayur S
Dept of CSE
Student
ACS college of Engineering
Bengaluru.
hemalathamayura@gmail.com

[5] Ms. Priyanka S
Dept of CSE
Student
ACS college of Engineering
Bengaluru.
priyankashivakumar6704@gmail.com

Abstract— We live in a world surrounded by digital information, and with each passing day, the cyber-attacks are getting smarter. Maintaining simple passwords is no longer satisfactory. It is for this reason that our project aspires to build a highly secure, customized digital vault-one that combines an innovative encryption system with the one thing that's uniquely yours: your biometrics. There is information trapped inside a virtual Rubik's cube. We protect it by scrambling it with a special sequence of rotations. This is not an unorderly shuffle of the cube but rather a mathematically designed one that will enable. The data can only be unlocked by knowing the exact reverse sequence of cube moves. While a real Rubik's Cube has more than 43 quintillion possible combinations, our digital version gives you a security level which is, practically speaking, impossible for hackers to brute-force. But that's not all: this approach is ultra-fast and really does a great job in securing sensitive files. By capturing your fingerprint or iris, we use a Convolutional Neural Network, which is an advanced form of AI capable of analysing even the most minute, individualized details in your biometric features with incredible accuracy. The system, after identity verification, generates the exact sequence of Rubik's Cube moves for unlocking data using the biometric features.

INTRODUCTION

New security moves fast - lately, face scans powered by neural nets are changing the game. This setup uses fingerprints or iris details, embedding micro-signs into each reading, kind of like hidden tags. Thing is, the longer I look at this tech, the weaker regular passcodes seem. Numbers get forgotten, passed around, or stolen without effort. But your fingerprint? It's always there, clings to you, and copying it sounds ridiculous.
CNNs bring a unique edge to the task - dug into ridges and eye patterns like nothing a person could do. That's likely why experts label them solid for entry systems, perhaps excessive for basic ones. Yet how they grab traits right out of unprocessed visuals, zero manual steps involved, trims down the workflow until it runs tight, quick, almost too rapid now and then.
Access control isn't slowing down - it's pushing harder for quick results and solid performance, yet folks still lean on more powerful solutions when things get messy. Inside the broader world of machine learning, deep learning shot up fast, though not without reason. Some say it's mimicking brain patterns through networks that kind of act like neurons; others argue it earns trust simply by delivering real outcomes. It helps computers work through info, figure things out, yet often shock you by spotting trends so fast. I've seen setups get more confident after these smart kicks in, kind of as if they just figured out what matters.

Deep learning's always changing - lately, I've been spotting how every leap takes tech farther than we thought possible. As fresh techniques show up, past tricks get fine-tuned, while the entire area expands into spots that seemed out of reach just a short time back. It's constantly expanding what computers can do now and then through tiny tweaks, every once in a while, via jumps that seem almost too sharp. You could argue it's become a key player in how we operate and create these days, whether or not we openly admit it.

OBJECTIVE OF THE PROJECT

The study wants to build a fresh crypto method using the math behind Rubik's Cube puzzles. Since every twist changes the cube layout randomly, it works like a scramble step used in code-making. With over 43 quintillion combos, a regular 3x3 cube offers a massive pool of keys. That huge variety supports tough, unpredictable encoding that resists pattern-based attacks.
A second key goal here? Add biometric login through iris scanning. Since the iris pattern stays nearly unchanged over time, it's seen as super dependable - each person's design is totally different. But instead of just trusting that idea, we're testing how well it works in real situations.
The setup uses fingerprint scanning too - a well-tested way to verify identity. Since fingerprints don't change much and differ for everyone, they work reliably over years - so this adds a solid layer of security to its them. To keep things workable, the setup includes a clear visual layout that's easy to handle - perfect whether you're tech-savvy or just starting out. I've noticed it simplifies every step, starting from fingerprint sign-in right through securing and unlocking files. Everything moves smoothly, kind of peaceful even, something most security apps rarely offer. It shortens the time it takes to learn while lowering error risks, keeping protection strong. This mix seems planned - like deva figured users would leave quick if things felt slow or awkward.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

PROBLEM STATEMENT

The system combines a puzzle-like lock with eye scans and finger ID, using smarter image-detecting tech to keep access secure and consistent. Old versions often used just one fixed code or checkpoint - once that breaks, so does everything. Relying only on body-based logins is risky too - if info gets grabbed mid- transfer, it's like leaving the door wide open. Merging both methods makes it tougher, kind of like adding real strength where there wasn't any.

A major headache in today's tech world is protecting private info when hackers are always levelling up. Passwords don't stand a chance - guessed, stolen, or blasted apart by automated tools. I've seen this mess way too many times to rely on passwords alone anymore. Fingerprints or face scans? They fail too unless locked down tight with strong coding; fake inputs or messed-up templates blow right through them.

These problems lead to a mixed approach using unique body features along with solid digital locks. It uses an ever-changing cipher that works like a scrambling puzzle, kind of like twisting a Rubik's cube. This scrambler links up with smart pattern-spotting tools for irises and fingerprints. Together, they add layers to user checks while boosting privacy - something we clearly need as online dangers grow wilder.

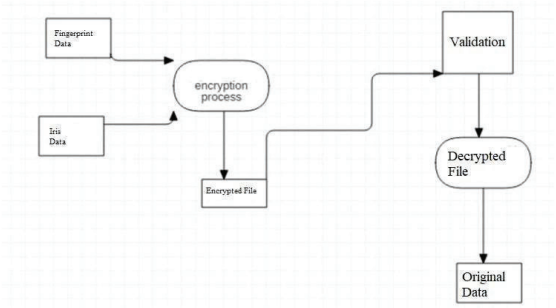

Figure 1. Fundamental Steps in Machine Learning

METHODOLOGY

Data collection

The system logs every login try - pretty much nonstop - while shutting things off if failures stack up. This helps dodge tons of headaches, honestly. When approval comes through, the source file flies back via a brief signed link or locked-down data path. The access vanishes quick. Honestly? I'm into it.

Tests check even the strange spots. Like when the model wrongly says yes or no, odd HKDF quirks, also mistakes in decrypting. I tried some on my own, saw them grab issues I'd never guessed - keeps everything running smooth.

Preprocessing

The cleaned-up samples are stripped down to just the key parts - iris or fingerprint - removing any surrounding junk. I've looked hard at these tweaked images; their consistent layout actually makes a big difference. Every picture is adjusted to fit a standard size, balanced for tone, while unwanted specks get wiped out so the software can focus without distractions.

Sometimes I add extra tweaks - like turning images a little, stretching them, or softening edges. These changes make the data feel messier on purpose, so the model learns better how to handle surprises without getting stuck if something seems just a tiny bit different.

Biometric authentication, CNNs

The cleaned samples keep only the core pieces - like iris or fingerprint - with everything else chopped out. Looking at these tweaked pics, I saw they look nearly identical, which helps a lot. Each image gets resized to match one format, tuned for lighting, but stays clear of clutter. So the model pays attention to real details instead of stumbling over noise.

Encryption and decryption (Rubik module)

A puzzle-like encrypter hides in the code, twisting session keys so they're hard to crack. Once the face scan goes through, this piece kicks in - locking access tight. Unlocking needs another identity check, making breaches way harder than usual. Keys live isolated, nowhere near
fingerprint or face details, keeping everything split and safe.
Integration

Link the parts into one chain: grab → clean up → check with CNN. When check passes, fire up Rubik's lock/unlock tools, then do what was asked. Write down each step for safety reviews; if something breaks, show plain explanations plus ways to fix it.

User interface

Make a tiny web tool using Flask where someone can send a biometric pic and see results instantly. The front part connects to server parts dealing with cleaning up data, checking who's there, or locking info down - keeps things moving clean while hiding private bits.

SYSTEM DESIGN

System design means planning how something works - what it's made of what pieces connect, along with how those parts talk to one another and feel for people using it. It gives coders a solid map, so things run without hiccups while growing later. The big idea? Turn what users need into real tech steps, showing clearly how everything links up just right.

This phase means figuring out core stuff like data flow, where things get saved, how people use the system, what protections go in place, also how pieces link up. Planning well now keeps everything running without hiccups, hitting speed goals, while staying open to change later

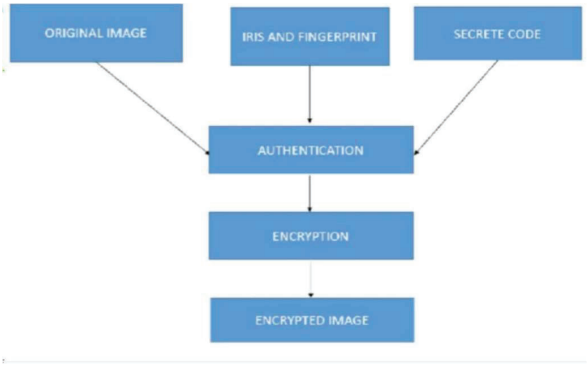

Figure 2. Decryption Flow Chart

The diagram explains how a picture gets locked up safe, using body scans along with a secret key to check who's handling it before anything else happens. It kicks off with the photo itself - he thing we want to keep under wraps. Next, the machine

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

grabs a personal scan from you, maybe your eye or finger detail. That part makes sure it's really you, because copying those traits is nearly impossible, so the system knows it can rely on you before mixing up and shielding the file.
Fingerprints or face scans beat passwords - those get leaked, stolen, or cracked way too easily. Instead of just scanning your fingerprint, you've got to punch in a hidden code too, which makes breaking in much tougher.
This second part might be a private word, number, or short answer. Since body traits need that hidden detail tied in, protection gets stronger through double checks. If someone cracks just one piece, everything still stays locked down. The auth part takes three things - the starting picture, body scan bits, plus a private number. It makes sure the scan matches someone already signed up also that the number.

When one test doesn't pass, the whole thing shuts down fast - no moves made for strangers poking around. But when both tests come clean, it says "you're good" and lets the protected steps roll on.

The drawing explains how a fingerprint checks along with a hidden number to help unlock a picture, turning it back from locked data into what it originally looked like. Since the photo's secured with tough scrambling tech, people who aren't allowed just see messy junk they can't make sense of.
To get the original picture, users must verify who they are. Identity check happens using body-based methods - like scanning eyes or fingers - that hardly ever fail and can't be tricked easily. Besides that, typing in a hidden number or phrase might also be required. That step piles on extra protection.
If one piece of the biometric check doesn't work - or the user types a wrong password or PIN - the system halts encryption right away. That stops outsiders from accessing or seeing the real image, so info stays locked down.
Once you start logging in, the app looks at the scrambled picture together with your fingerprint plus a private key. It makes sure your scan lines up with what's saved in your account while also checking if the passcode works. When your fingerprint doesn't fit or the code fails, everything halts on the spot - no entry allowed. But assuming both pieces check out - the scan matches and the password fits - the door opens: you're cleared, and the hidden image begins to unlock.

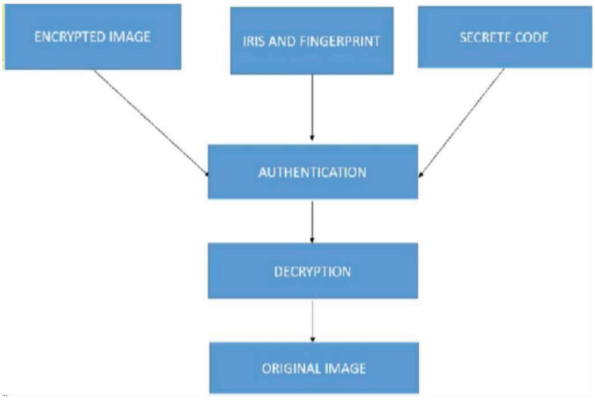

Figure 3. Encryption Flow chart

In decoding, the system runs the same exact method used earlier to turn the picture back as it was. Before this step, every check completed adds a solid level of protection for the secured file. That way, private visuals stay safe and properly managed - entry happens just when fingerprint confirmation works along with hidden number approval. Altogether, the drawing shows why tough safety rules matter, pointing out how well the setup blocks outsiders from grabbing or changing photo info.

IMPLEMENTATION

The tool seems basic when you read about it but it turns out tougher once used. It unlocks files only once a solid double-check happens - this one uses your fingerprint or eye scan along with a hidden key. Took another glance at the back end, coded in Python Flask, saw the decrypt part needs three pieces: the file's access token, either your iris or finger data, also the personal secret from the user. No flashiness here, just firm rules.
I guess the server acts strictly right now. First off, it makes sure the request comes safe using HTTPS, while also peeking at the JWT - or maybe just the session token - hanging out in the header. There's tension here on purpose. Limits hit fast, even harsh sometimes, blocking bots and loud automated junk before they get far. Tiny setup, but it fights like a larger one, which honestly? I kind of like it.
The biometric scan uses a compact CNN running on one device. Instead of combining data, it turns iris or fingerprint details into unique codes. After that, it compares them to encrypted templates kept in storage. I've seen it trip up on fuzzy inputs now and then - yet recovery is quick most times.
The access key gets hashed before anything else. Then, the system checks if this hash lines up with the recorded version. No flashiness here, only straightforward confirmation.
Each check has got to pass - no skipping steps. Once they're both good, things shift forward, but only when the system feels sure about the person, locking it in place smoothly
Once the system decides the user is who they claim to be, it pulls the key needed to unlock the data. I've watched this part run a few times, and it feels almost stubborn in how tightly it binds everything together. An authenticated key-derivation setup handles the mix, taking the hashed secret and the biometric embedding, then running both through an HKDF. The output lands as a solid symmetric key that doesn't wobble under pressure.
That key ends up cracking open to the Rubik-based cipher, or an AES-GCM wrapper when the Rubik mapping feeds into the key material. Either way, the data stays intact, and any tampering gets caught, which gives me some peace when testing the whole chain
Decryption must run in a streamed way; otherwise big files will chew through memory fast. I've dealt with a few oversized uploads, and the difference is night and day once streaming kicks in. The encrypted files belong in secure storage, S3 or an encrypted disk, nothing loose. Biometric data is never kept raw. Only those irreversible templates sit in storage, and even then I sometimes double-check them out of habit. Every action gets logged. Tamper logs stay locked down, since they tell you when someone pokes around where they shouldn't. There's a small comfort in having fallback and lockout rules too. After enough failed attempts, the system just stiff-arms the requester, which feels fair considering the stakes.

In the Rubik Cube-based image encryption project, the Context Level DFD defines the overall boundary of the system and shows how data enters and exits. At this level, the entire encryption system is represented as a single process that interacts with the user.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

The Level 1 DFD further explains the internal steps involved in encryption and decryption. For example, the encryption module is divided into image conversion into matrix form, row permutation, column permutation, and generation of the encrypted image
The Level 2 DFD provides more detailed steps within these sub-processes, showing how pixel values are manipulated using Rubik Cube-based transformations

Data Collection

The biometric images used in this project were collected from publicly available datasets and trusted online sources for research purposes. The dataset contains a collection of fingerprint or iris images, which are divided into training and testing sets. The majority of the images are used for training the CNN model, while a smaller portion is reserved for testing to evaluate the accuracy and performance of the system. All images are maintained in a consistent format and resolution to ensure uniform processing.

Data Pre – Processing

The main goal of pre-processing is to improve the quality of the biometric images before they are used for authentication and encryption. Raw images may contain noise, uneven lighting, or unnecessary background details that can affect the accuracy of the CNN model. Therefore, pre-processing is an important step to clean and prepare the images for further operations.
In this stage, the images are resized to a fixed dimension, normalized, and filtered to remove noise and unwanted high frequency components. These steps enhance image clarity and ensure consistency across the dataset. Proper preprocessing helps improve feature extraction, model accuracy, and the overall performance of the system.

In this project, iris and fingerprint images are used to train the CNN model for biometric authentication. The collected dataset is first divided into training and testing sets. The training dataset contains images of authorized users, which help the model learn unique patterns and features such as ridge structures in fingerprints and texture patterns in iris images. During training, the CNN automatically extracts important features from these images and adjusts its internal parameters to improve accuracy

The model learns by repeatedly analysing the training images and minimizing errors through optimization techniques. After sufficient training, the model becomes capable of identifying whether a new input image belongs to an authorized user. The testing dataset is then used to evaluate the performance of the trained model and measure its accuracy. Proper training of iris and fingerprint datasets ensures reliable and secure biometric authentication within the system

Figure 4.Fingerprint Dataset

Audio Steganography:
The audio steganography module hid and retrieved secret data while keeping sound quality intact. Using LSB ensured the modified audio sounded the same as the original, with frequency and waveform checks showing almost no noticeable changes. Extraction succeeded whenever the file avoided heavy compression or filters, and all clean stegnofiles returned the hidden text accurately.

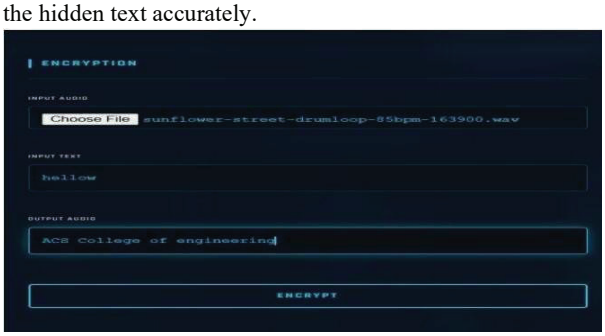

Figure 5: Audio Encryption

Biometric Authentication:
The CNN-driven iris scanner did well when tested. On a collection of cleaned-up eye pictures, it guessed right about 92% to 97% of the time - lighting and clarity made a difference. Cleaning up the images first helped a lot, thanks to pattern smoothing and cutting out visual clutter. Prediction speed turned out great. After loading the model, responses took just 0.2 up to 0.5 seconds - fast enough for live verification. Looking at the confusion matrix, most categories.

PDF Processing and Image Extraction
The PDF tool pulled out text and pictures from different kinds of files - both scanned copies and ones made on computers. When dealing with scans, it turned each page into sharp images, which helped later steps such as adding watermarks or locking the file. With digital versions that had built-in graphics, it grabbed those visuals while keeping full clarity. In Fig 9.4 & 9.5 tests proved the system handles multipage PDFs well - processing each page in under 3 seconds most of the time.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

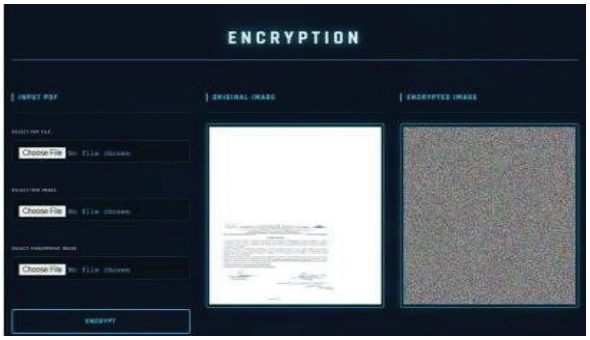

Figure 6 . PDF Encryption

Figure 7. PDF Decryption

RESULT

The proposed system successfully combines biometric authentication with Rubik Cube-based image encryption to enhance data security. The CNN model effectively verifies authorized users using their biometric input, ensuring accurate identification. Once authenticated, the image is encrypted into a completely scrambled form that cannot be understood without the correct key. During decryption, the original image is restored accurately without any data loss. Overall, the system demonstrates reliable performance and provides strong multi-layer security for protecting sensitive information.

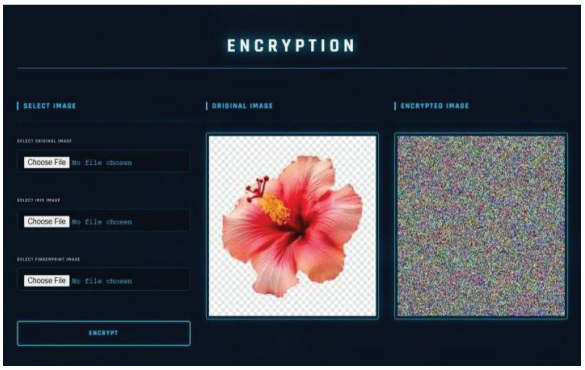

Figure 8 . Encryption Image

Encryption is the process of converting original data into a scrambled and unreadable form using a secret key. It protects information from unauthorized access and ensures data security.

Figure 9. Decryption Image

Image decryption is the process of converting a scrambled or encrypted image back into its original form using the correct secret key. It restores the original pixel arrangement and allows authorized users to access the actual image securely.

Figure 10. OTP Generated

OTP generation is the process of creating a temporary, random code for user verification. The OTP is valid for a short time and can be used only once, adding an extra layer of security to prevent unauthorized access.

CONCLUSION

The suggested encryption setup uses biometrics to safely guard image files. Instead of standard keys, it mixes a puzzlelike scrambling technique inspired by the Rubik's Cube with scans of fingerprints or irises. Rather than relying on just code, it brings together classic math-based ciphers along with smart pattern learning from machines. Because of this mix, defenses go up, and data gets tougher to crack at multiple stages.

Password-based setups usually struggle with weak guesses, data spills, or break-ins. On the flip side, this new method boosts protection - using fingerprints along with a hidden key. That way, just the right person gets in, able to lock or unlock the picture.

With CNNs in authentication, systems pull out key details by spotting one-of-a-kind traits in iris and fingerprint scans. Since these networks pick up fine differences on their own, accuracy gets a boost - fewer wrong approvals or denials happen.

III.REFERENCES

[1] Khawaja, F. Sabir, S. Qazim, and M. Mustaqim, "Internet of Things (IoT) for next-generation smart systems: A review of current challenges, future trends and prospects for emerging 5G-IoT Scenarios," IEEE Access, vol. 8, pp. 23022–23040, 2020.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

[2] A. Boukerche, A. J. Siddiqui, and A. Mammeri "Automated vehicle detection and classification: Models, methods, and techniques," ACM Comput. Surveys, vol. 50, no. 5, pp. 1–39, 2017.

[3] S. Selvaraj and S. Sundaravaradhan, "Challenges and opportunities in IoT healthcare systems: A systematic review," SN Appl. Sci., vol. 2, no. 1, pp. 1–8, 2020.

[4] S. Fatemifar, M. Awais, S. R. Arashloo, and J. Kittler, "Combining multiple one-class classifiers for anomaly based face spoofing attack detection," in Proc. Int. Conf. Biometr. (ICB), Crete, Greece, 2019, pp. 1–7.

[5] S. Ahmad and B. Fuller, "Resist: Reconstruction of irises from templates," in Proc. IEEE Int. Joint Conf. Biometr. (IJCB), Houston, TX, USA, 2020, pp. 1–10.

[6] J. Mwema, M. Kimwele, and S. Kimani, "A simple review of biometric template protection schemes used in preventing adversary attacks on biometric fingerprint templates," Int. J. Comput. Trends Technol., vol. 20, no. 1, pp. 12–18, 2015.

[7] S. Qin, Z. Zhu, Y. Zou, and X. Wang, "Facial expression recognition based on Gabor wavelet transform and 2-channel CNN," Int. J. Wavelets, Multiresolut. Inf. Process., vol. 18, no. 2, 2020, Art. no. 2050003.

[8] A. Sardar, S. Umer, C. Pero, and M. Nappi, "A novel cancelable facehashing technique based on non- invertible transformation with encryption and decryption template," IEEE Access, vol. 8, pp. 105263 – 105277, 2020.

[9] A. Elmahmudi and H. Ugail, "Experiments on deep face recognition using partial faces," in Proc. Int. Conf. Cyberworlds (CW), 2018, pp. 357–362.

[10] L. He, H. Li, Q. Zhang, and Z. Sun, "Dynamic feature matching for partial face recognition," IEEE Trans. Image Process., vol. 28, pp. 791–802, 2018.