Unleashing Palm Vein Recognition via Smooth Mask Mixup and Large Kernel-based Network

Xin Jin*, Hongyu Zhu*, Mounim A. El-Yacoubi, *Senior Member, IEEE*, Haiyang Li, Xingrong Fan, Lin Chen, Yun Jiang†, Qun Song†

***Abstract*—Vein Recognition represents a new generation of biometrics, offering high security and convenience. However, existing convolutional neural networks (CNNs) for palm vein recognition are often constrained by the limited effective receptive field (ERF) and insufficient training data, which restrict their ability to model global vascular dependencies and structural patterns. To address these challenges, we propose StarLKNet, a large-kernel convolutional architecture tailored for palm vein recognition, jointly with a structure-aware augmentation strategy termed StarMixup. StarLKNet integrates large and small-kernel convolutions with a channel gating mechanism to enable adaptive multi-granularity feature fusion. Furthermore, StarMixup introduces Gaussian-based smooth masks that preserve the sparse and continuous topology of vein patterns during sample mixing, thereby expanding the training distribution while maintaining vascular structural integrity. Extensive experiments are conducted on four public palm vein datasets, evaluating comprehensive metrics including recognition accuracy, equal error rate (EER), and TPR@FPR. Comparisons against a wide range of baselines, including small-kernel CNNs, vision transformers, and some vein-specialized models, demonstrate that StarLKNet consistently achieves state-of-the-art accuracy and the lowest EER, while StarMixup provides competitive augmentation gains.**

***Index Terms*—Data Augmentation, Palm Vein Recognition, Biometrics, Computer Vision.**

I. Introduction

WITH the growing emphasis on personal information security in modern society, extensive studies have been carried out on biometric technologies recently, which authenticate individuals via unique physiological features (e.g., face [1], fingerprint [2], vein [3]) or behavioral patterns (e.g., gait [4], eye movement [5]). However, these external features, such as the most widely used fingerprints and facial features, are increasingly susceptible to sophisticated forgery attacks [6]. Conversely, vein recognition is a superior alternative, leveraging internal vascular structures that are impervious to external factors (e.g., skin moisture and wear) and inherently enable liveness detection by the exclusive presence of deoxygenated hemoglobin in living circulatory systems.

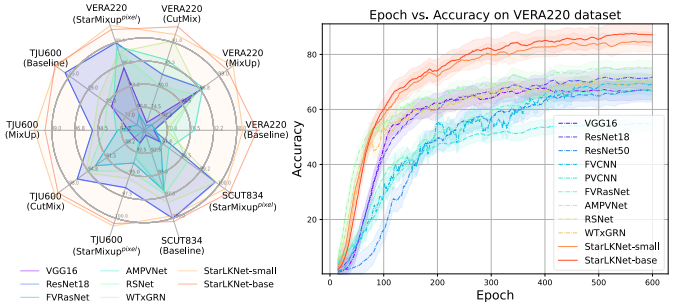


Fig. 1. ***Left:*** Model performance comparison across VERA220, TJU600, SCUT834 under baseline and augmentations (MixUp, CutMix, StarMixup$^{\text{pixel}}$; ***Right:*** Training accuracy curves of models on VERA220 over 600 epochs without augmentation, StarLKNet-base/small outperforms others.

In vein recognition, traditional approaches primarily rely on handcrafted feature extraction, coupled with shallow machine learning classifiers [7], [8]. These methods presuppose that vein patterns manifest as valley or line-like shapes, thereby overlooking diversity, higher-dimensional information inherent in vascular structures. In contrast, deep learning paradigms enable end-to-end feature extraction without prior assumptions, but they require large datasets to optimize network parameters. However, stringent privacy regulations, ethical concerns, and storage limitations severely restrict the availability of sufficient samples in data collection. Such data scarcity intensifies overfitting, making high-performance and robust models hard to achieve, while also obstructing the deployment of vein recognition in real-world, security-critical applications.

To mitigate the limitations arising from limited training data, researchers have proposed various Data Augmentation (DA) methods to generate new samples and expand the training set. Among these, mixup-based techniques [9]–[11] have gained popularity due to their simplicity and effectiveness across diverse tasks. However, vein images are inherently continuous and sparse, so their features differ fundamentally from those of general natural images. Therefore, applying generic mixup methods to veins may lead to suboptimal performance. For instance, CutMix [10] replaces pixel-wise mixing with patch-wise mixing. Its random patch selection and pasting can disrupt the continuity of vein patterns. This disruption leads to feature dropout, thereby hindering the ability of the encoder to learn robust representations, as illustrated in Fig. 3. This continuous and sparse distribution also poses a broader architectural challenge for vein recognition. Models built with small-kernel convolutions, which have a limited *Effective Receptive Field* (ERF), struggle to capture the continuous, sparse dependencies

Xin Jin is with the Chongqing Technology and Business University, Westlake University, China. E-mail: jinxin86@westlake.edu.cn.

Hongyu Zhu and Lin Chen are with the Chongqing Institute of Green Intelligent Technology, Chinese Academy of Sciences; Chongqing School, University of Chinese Academy of Sciences, Chongqing, China. E-mail: zhuhongyu@cigit.ac.cn.

Mounim A. El-Yacoubi is with SAMOVAR, Telecom SudParis, Institute Polytechnique de Paris, 91120 Palaiseau, France. E-mail: mounim.el_yacoubi@telecom-sudparis.eu.

Haiyang Li, Xingrong Fan, Qun Song, and Yun Jiang are with the School of Artificial Intelligence, Chongqing Technology and Business University, Chongqing, China. E-mail: lihaiyang@ctbu.edu.cn, fanxingrong@ctbu.edu.cn, songqun@ctbu.edu.cn, jiangyun@ctbu.edu.cn.

Xin Jin and Hongyu Zhu contributed equally to this work.

(Corresponding authors: Qun Song and Yun Jiang.)

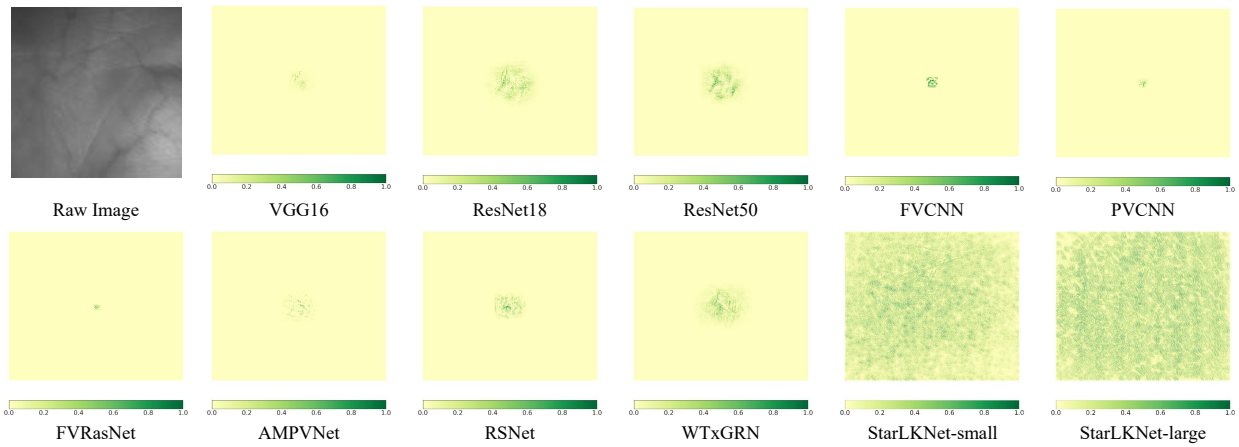


Fig. 2. ERF visualization comparison across 11 models. A more widely distributed dark area indicates a larger ERF. The raw input image is shown on the left, with corresponding ERF heatmaps (scaled 0–1) for each model. StarLKNet-small and StarLKNet-large demonstrate significantly broader ERF coverage compared to baseline models (e.g., VGG16, ResNet18) and other custom architectures.

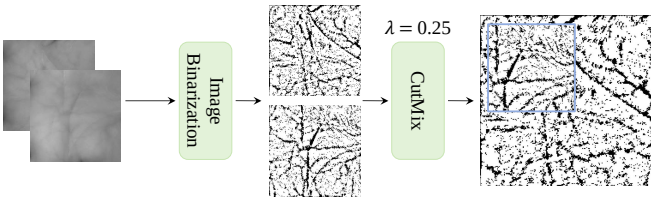


Fig. 3. Visualization of CutMix for the vein feature distribution. The cutting patch is noted with the blue bounding box. We applied mean adaptive thresholding to binarize the image, then used CutMix to verify that the vein features were corrupted.

inherent to vein features, thereby limiting their [12]–[14] performance. Recent studies [15], [16] have indicated that Convolutional Neural Networks (CNNs) with a larger ERF can rival the performance of Vision Transformers (ViTs) [17]–[19]. In line with this, we find that CNNs with large kernels are significantly more adept at capturing vein feature distributions. As demonstrated in Fig. 1, the large kernel exhibits faster fitting speeds and higher recognition than the small-kernel framework represented by ResNet18 [20].

In this work, we build upon our prior research [21]. First, we propose StarMixup, a tailored mixup strategy that generates smooth masks by a Gaussian function, better preserving the texture features of the palm vein. Our strategy adaptively and dynamically selects between StarMixup and vanilla MixUp [9] based on a threshold, thereby synergistically combining their benefits. Central to our framework is StarLKNet, a novel architecture that integrates multi-granularity feature extraction via a dual-kernel design, in which large kernels capture long-range spatial dependencies essential for modeling the sparse, continuous topology of venous networks, while small kernels preserve fine-grained local details. To further refine feature representations, we equip it with a ChannelGating Neck that decouples features into global and gating streams, enabling adaptive recalibration of salient information. Crucially, the structure of StarLKNet substantially expands the ERF, as visualized in Fig. 2.

Experimental results show that the StarLKNet-based model, compared with existing small-kernel CNNs, ViTs, and vein-specific models, achieved significant improvements in recognition accuracy and Equal Error Rate (EER), with notable gains at low False Positive Rates (FPR) on four public palm vein datasets (TJU600, SCUT834, VERA220, and CASIA200). The proposed StarMixup augmentations further boost performance beyond existing mixup methods. Ablation studies confirm the synergistic effect of dilated convolutions and gating. These results demonstrate the strong potential of StarLKNet and StarMixup for high-security palm vein recognition.

In summary, our main contributions are as follows:

- We rethink the role of convolution kernel size in palm vein recognition and reveal that enlarging the ERF is critical for capturing the vein features, yielding substantial performance gains.
- We propose StarLKNet, a large kernel-based architecture that integrates multi-scale convolutions with a channel gating mechanism for adaptive multi-granularity feature extraction, achieving new state-of-the-art results.
- We introduce StarMixup, a structure-aware DA strategy that generates Gaussian-based masks to preserve the topological integrity of vein patterns during mixing, thereby mitigating overfitting and improving generalization.
- We provide the most comprehensive empirical study to date in palm vein recognition, evaluating four public datasets with multiple metrics and extensive baselines ranging from CNN- and ViT-based to vein-specialized models, establishing a solid benchmark for future research.

II. Related Work

A. Palm Vein Recognition

Due to the property of hemoglobin absorbing near-infrared light, infrared cameras are capable of capturing clear vein images [14]. However, the unique and intricate vein distribution patterns present significant challenges for effective feature extraction. To address these issues, research has generally followed two paradigms: 1) Shallow machine learning-based approaches [7], [8], [22], [23], which are increasingly being phased out due to their heavy reliance on hand-crafted features and insufficient robustness against complex backgrounds or alignment variations; 2) CNN-based approaches, which

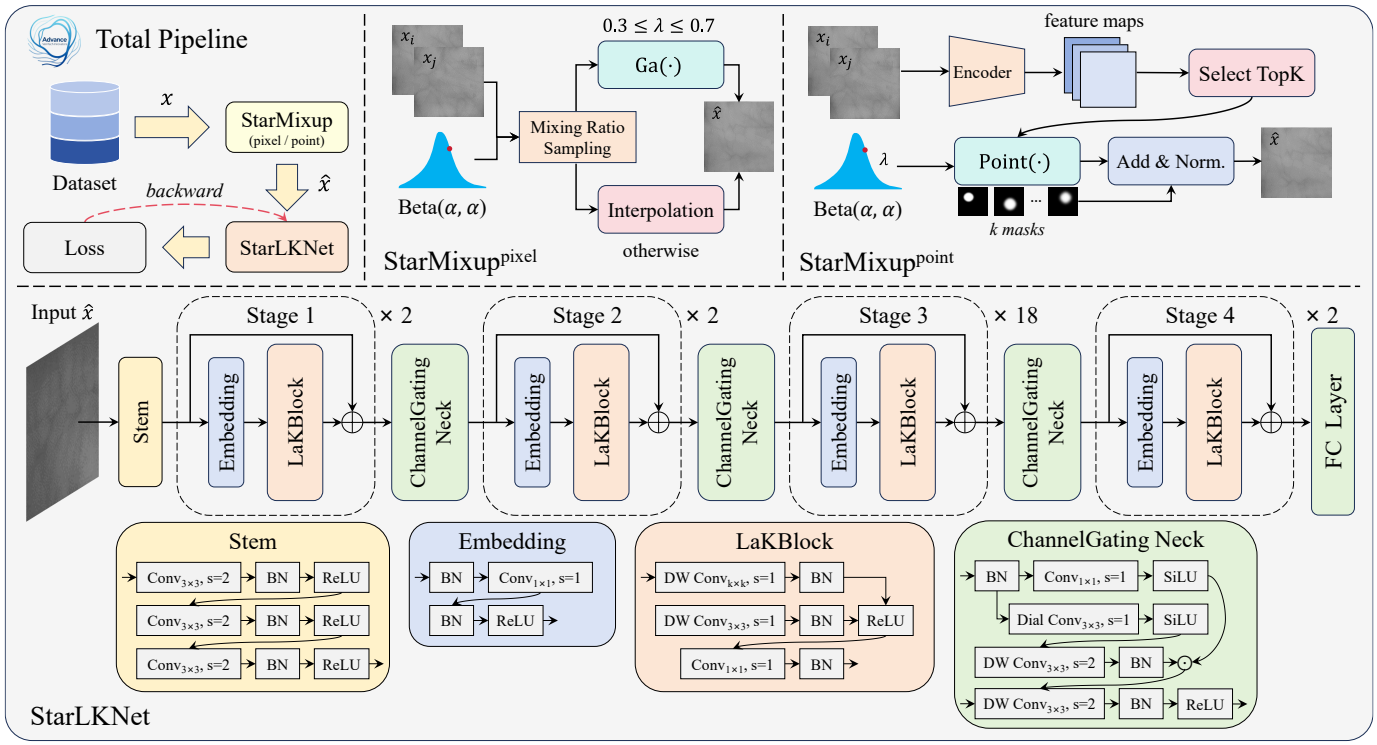


Fig. 4. Overall pipeline of our proposed StarMixup augmentation strategy and StarLKNet backbone. ***Top section:*** The overall training pipeline, where input samples x are augmented by either StarMixup$^{\text{pixel}}$ or StarMixup$^{\text{point}}$ to generate $\hat{x}$, which is then fed into StarLKNet for training. ***Bottom section:*** The detailed architecture of StarLKNet, which consists of a Stem, four stages (with repeated LaKBlock and ChannelGating Neck modules), and a final FC layer.

automate deep feature extraction by exploiting powerful non-linear representation capabilities to achieve more accurate and discriminative identification. In the evolution of CNNs, Yang et al. [13] proposed FVRAS-Net, integrating recognition and anti-spoofing via multi-task learning. To address data scarcity, Qin et al. [14] introduced PVCNN, utilizing GAN-based augmentation to overcome single-sample (SSPP) bottlenecks. Shen et al. [24] developed the lightweight LWCNN to optimize real-time matching efficiency. For unconstrained environments, Luo et al. [25] presented AMPVNet, ensuring high robustness against pose and grayscale variations. Most recently, Luo et al. [26] proposed RSNet, employing a dual-branch structure to deepen the discriminative extraction of region-specific physiological features; Qin et al. [27] further proposed WTxGRN, which employs wavelet transforms and an extended gated recurrent architecture (xGRU) for parallel extraction of global and local features, while introducing a spiking version (Spiking WTxGRN) to significantly enhance energy efficiency for real-time recognition on edge devices.

B. Mixup Augmentations

The inception of mixing-based DA methods began with MixUp [9], which generates augmented samples by performing linear interpolation between two or more images and their corresponding labels using a mixing ratio. This simple yet effective strategy encourages smoother decision boundaries and has been shown to improve generalization across various tasks, including super-resolution [28], segmentation [29], regression [30], and long-tail distribution learning [31]. CutMix [10] extends MixUp from the pixel level to the spatial level by randomly generating a mask of the same size to combine image patches. Subsequently, various approaches have been proposed to enhance strategies for mixing samples or labels [32]–[35]. Compared to manually designed methods, SaliencyMix [11] leverages an additional feature extractor to obtain saliency information, enabling more guided and meaningful sample combination. With a similar objective, PuzzleMix [36] and Co-Mix [37] leverage gradient information during backpropagation to identify salient feature regions and apply an optimal transport strategy to prevent overlap of informative features in the mixed samples, ensuring effective feature maximization. AutoMix [38], on the other hand, employs an end-to-end framework that introduces a dedicated mix block that jointly optimizes the mixing generator and the model, thereby achieving a favorable balance between computational efficiency and performance. More recently, AdAutoMix [39], built upon AutoMix, has been proposed to augment the generated samples by mixing any set of N samples (not just two) and to use adversarial training to prevent generator overfitting by creating difficult samples that have a greater impact on training performance. With the powerful generative model DiffuseMix [40], a pipeline is built that uses a Diffusion Model and a text prompt to generate augmentation, which can be done manually. Jin et al. propose MergeMix [41] to trade-off the performance and efficient for image classification via token merging on ViT-based models.

III. Methodology

A. Overview

Our proposed model is illustrated in Fig. 4 and comprises two components: StarMixup and StarLKNet. StarMixup employs a

soft masking-based augmentation to generate suitable samples, whereas StarLKNet achieves multi-granularity feature extraction via a large kernel-based CNN with a gating mechanism that decouples features into global and gating streams.

B. Preliminaries

Mixup formulations. We define $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ be the training dataset, where $x_i \in \mathbb{R}^{h\times w\times c}$ denotes the i-th input image with height h, width w and channels c, and $y_i \in \{0,1\}^K$ is the corresponding one-hot encoded ground-truth label (K is the number of classes). Mixup augmentation is mainly divided into two cases: interpolating-based and masking-based. An interpolation-based method like MixUp [9] creates virtual training samples by linearly interpolating pairs of samples and their labels. For two selected samples (x_i, y_i) and (x_j, y_j), the mixed sample $\hat{x}$ and label $\hat{y}$ are generated as:

$$\begin{aligned} \hat{x} &= \lambda \odot x_i + (1-\lambda) \odot x_j, \\ \hat{y} &= \lambda * y_i + (1-\lambda) * y_j, \end{aligned} \tag{1}$$

where the ratio λ sampling from a Beta(α, α) distribution.

Masking-based methods such as CutMix [10], by contrast, generate mixed samples using a binary mask $\mathcal{M} \in \{0,1\}^{h\times w}$, which is generated by metrics such as attention scores, gradients, or saliency maps. Then, the mixed samples were obtained as:

$$\hat{x} = \mathcal{M} \odot x_j + (1-\mathcal{M}) \odot x_i, \tag{2}$$

where $\odot$ denotes the element-wise product. Let $f_\theta(\cdot)$ be the deep neural network parameterized by θ. Both MixUp and CutMix train the model by minimizing the loss between the prediction $f_\theta(\hat{x})$ and the mixed label $\hat{y}$. The final optimization object is a Mixup Cross Entropy loss as in Eq. (3):

$$\mathcal{L}_{\text{MCE}} = \lambda * \mathcal{L}_{\text{CE}}\big(f_\theta(\hat{x}), y_i\big) + (1-\lambda) * \mathcal{L}_{\text{CE}}\big(f_\theta(\hat{x}), y_j\big). \tag{3}$$

Gating module. Dynamic feature modulation has been widely adopted in vision architectures to enhance representation learning through adaptive reweighting mechanisms [42]. In this work, we introduce a lightweight gating module composed of a 1×1 convolution ($\text{Conv}_{1\times1}$) followed by the SiLU activation. The 1×1 convolution performs channel-wise linear projection, enabling feature recalibration across channels with negligible computational overhead. Subsequently, the SiLU function introduces smooth non-linearity and input-dependent modulation. The SiLU activation is defined as:

$$\text{SiLU}(x) = x \cdot \frac{1}{1+e^{-x}}, \tag{4}$$

which can be interpreted as a self-gating operation, where each feature is scaled by its sigmoid response. Compared to ReLU, SiLU maintains non-zero gradients for negative inputs and provides smoother transitions and more stable optimization.

Given an input feature map $z \in \mathbb{R}^{w\times h\times c}$, the gating operation is defined as:

$$\text{Gating}(z) = \text{SiLU}\big(\text{Conv}_{1\times1}(z)\big). \tag{5}$$

This formulation can be viewed as an implicit multiplicative modulation mechanism, where the convolution learns channel-wise importance coefficients, and the SiLU activation adaptively scales features in a data-dependent manner. Unlike heavy attention modules, this design introduces minimal parameters while retaining the ability to suppress irrelevant responses and amplify discriminative patterns.

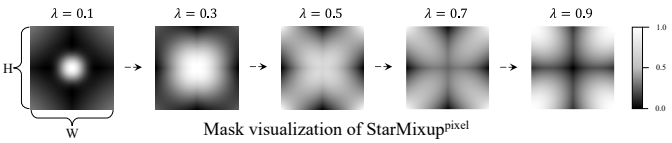


Fig. 5. From *left* → *right* are the visualizations of StarMask with different λ.

C. StarMixup Augmentation

This subsection details the StarMixup$^{\text{pixel/point}}$, the pseudo code are provided in Algorithm 1 and Algorithm 2.

a) StarMask generation: As illustrated in Fig. 5, the proposed StarMask strategy, compared to the traditional mixing ratio λ, places greater emphasis on both the central region and surrounding areas of the samples, demonstrating better adaptability to the spatial distribution characteristics of vein patterns. To obtain StarMask, we define a Gaussian function Ga$(\cdot)$ to generate the mask $\mathcal{M}$. We assume a pair of samples (x_i, y_i), (x_j, y_j), with data $x \in \mathbb{R}^{w\times h\times c}$, and sampling λ from Beta(α, α). The Gaussian function is given by Eq. (6):

$$\text{Ga}(x, y) = \exp\big(-\frac{(x-\frac{k}{2})^2 + (y-\frac{k}{2})^2}{2\sigma^2}\big), \tag{6}$$

where k is the size of the Gaussian kernel set to the height of input samples when the network is a convolution-based classifier. It is set to patch size when the network is a transformer-based classifier, then upsampled to the height of input samples by scaling, since ViTs need to be patchified, which differs from pixel-level masks, and converted to the token level, $\sigma = \lambda * h$. Passing the parameters to Ga$(\cdot)$, we obtain the mask $\mathcal{M}$ as Eq. (7):

$$\mathcal{M} = \frac{1}{N}\sum_{n=1}^{N} \text{Ga}(k_n, h, \sigma_n), \tag{7}$$

where $N = 3$. Note that $\sigma_2 = (1-\lambda) * h$, $\sigma_1 = \sigma_3$, and $k_3 = 2 * k_1$. Then, we normalize the aggregated mask to ensure its values lie within $[0, 1]$.

After getting the mask $\mathcal{M}$, we need to recalculate the correct mixing ratio $\hat{\lambda}$ according to Eq. (8):

$$\hat{\lambda} = \frac{\sum_{i=1}^{w}\sum_{j=1}^{h} \mathcal{M}_{i,j}}{w \times h}, \tag{8}$$

where $\mathcal{M}_{i,j}$ denotes the pixel value at the *i*-th row and *j*-th column in the mask. The value $\mathcal{M}_{i,j}$ can be viewed as the average intensity of all pixels in $\mathcal{M}$. Finally, the $\hat{x}$ and the corresponding label $\hat{y}$ are obtained according to Eq. (9):

$$\begin{aligned} \hat{x} &= \mathcal{M} \odot x_i + (1-\mathcal{M}) \odot x_j, \\ \hat{y} &= \hat{\lambda} * y_i + (1-\hat{\lambda}) * y_j. \end{aligned} \tag{9}$$

TABLE I
TOP-1 ACCURACY (%)↑ AND EER (%)↓ OF BASELINE, MIXUP, CUTMIX, SALIENCYMIX, AND STARMIXUP ON TJU600, ACC. MEANS ACCURACY.

TJU600	Param. (M)	FLOPs (G)	Baseline		MixUp		CutMix		SaliencyMix		StarMixup^{pixel}		StarMixup^{point}	
			Acc. ↑	EER ↓	Acc. ↑	EER ↓	Acc. ↑	EER ↓	Acc. ↑	EER ↓	Acc. ↑	EER ↓	Acc. ↑	EER ↓
Small-Kernel CNN Models														
VGG16$_{ICLR'2014}$	136.72	15.50	79.38	3.32	91.22	1.14	86.38	2.18	86.45	2.03	91.15	1.09	90.80	1.44
ResNet18$_{CVPR'2016}$	11.48	1.82	91.32	1.03	94.03	0.93	94.32	0.65	95.15	0.45	95.22	0.62	96.47	0.42
ResNet50$_{CVPR'2016}$	24.74	4.12	88.77	1.48	94.53	0.60	93.85	0.75	93.72	0.72	94.90	0.64	94.93	0.63
ViT-based Global Receptive Field Models														
ViT-small$_{ICLR'2021}$	48.47	9.42	–	–	81.98	2.34	71.35	4.03	71.90	3.57	79.55	2.82	79.27	2.78
ViT-base$_{ICLR'2021}$	86.26	16.86	–	–	83.23	2.28	72.63	3.73	73.32	3.83	81.13	2.84	80.22	2.66
DeiT-tiny$_{ICML'2021}$	5.64	1.08	–	–	79.97	2.68	67.48	5.90	67.23	5.58	77.93	2.94	74.23	3.63
DeiT-small$_{ICML'2021}$	21.90	4.24	–	–	81.93	2.51	74.68	3.31	75.12	3.36	81.71	2.54	81.10	2.26
SwinT-tiny$_{ICCV'2021}$	27.98	4.36	–	–	92.37	0.88	82.50	1.72	82.97	1.92	92.15	0.88	87.40	1.53
SwinT-small$_{ICCV'2021}$	49.30	8.52	–	–	92.35	0.88	83.32	1.89	84.10	1.75	91.87	0.96	87.82	1.36
SwinT-base$_{ICCV'2021}$	87.36	15.14	–	–	93.15	0.78	85.48	1.50	84.85	1.63	92.77	0.90	92.82	0.89
Vein-Specialized CNN Models														
FVCNN$_{TIFS'2019}$	24.88	48.31	62.68	9.32	87.37	2.34	75.72	4.08	78.52	3.34	88.30	2.05	87.63	2.22
PVCNN$_{TIFS'2019}$	4.49	5.45	69.65	5.14	90.30	1.78	78.18	3.80	79.67	3.45	90.67	1.40	89.40	1.95
FVRasNet$_{TIM'2020}$	0.89	0.76	72.88	3.52	92.08	1.17	91.77	1.69	92.03	0.96	92.98	1.08	91.97	1.11
AMPVNet$_{TIFS'2024}$	1.38	0.17	78.92	3.15	92.18	1.25	85.52	2.13	85.62	2.15	92.47	1.08	92.40	1.30
RSNet$_{TIFS'2025}$	2.33	0.18	80.48	2.45	93.13	0.90	92.77	1.60	90.20	1.54	93.73	0.86	93.67	0.78
WTxGRN$_{TIFS'2025}$	17.79	4.50	82.72	2.58	93.93	0.88	93.12	0.80	95.45	0.60	94.37	0.85	94.53	0.93
Large-Kernel Models (Ours)														
StarLKNet-tiny	3.18	1.12	92.42	1.53	95.73	0.30	96.75	0.44	96.18	0.56	96.58	1.31	97.68	0.27
StarLKNet-small	12.56	4.12	93.58	**1.53**	**98.55**	**0.16**	**97.40**	**0.37**	**97.33**	**0.44**	**98.70**	0.22	**98.78**	**0.18**
StarLKNet-base	42.06	14.42	93.93	1.79	97.73	0.21	96.98	0.57	96.37	0.48	98.48	**0.18**	95.87	2.96
StarLKNet-large	150.60	52.42	**94.70**	1.63	97.30	1.14	96.33	0.70	96.37	0.81	97.98	0.25	97.68	0.32
Gains	–	–	**+3.38**	**+0.50**	**+4.02**	**-0.44**	**+3.08**	**-0.28**	**+1.88**	**-0.01**	**+3.48**	**-0.44**	**+2.31**	**-0.24**

b) Threshold setting: As shown in Fig. 5, due to Ga(·), the obtained masks are not reliable when λ is less than 0.3 or more than 0.7, so we employ a threshold filtering strategy to avoid obtaining unreliable masks when mixing:

$$\hat{x} = \begin{cases} \mathcal{M} \odot x_i + (1 - \mathcal{M}) \odot x_j, & \text{if} \quad 0.3 \leq \lambda \leq 0.7, \\ \lambda * x_i + (1 - \lambda) * x_j, & \text{otherwise.} \end{cases} \tag{10}$$

We only employ the StarMask if λ is within the range of [0.3, 0.7]; otherwise, we choose interpolation. This threshold setting avoids the disadvantages of StarMixup^{piexl} in special cases.

c) Forward-based StarMixup: To further improve the adaptiveness and robustness of StarMixup^{pixel}, we propose a forward-based variant, named StarMixup^{point}, which generates the mask according to the gradient saliency of each sample.

Unlike StarMixup^{pixel}, where the Gaussian mask is constructed using a predefined spatial prior, StarMixup^{point} first performs a forward and backward pass to obtain the gradient map $\mathcal{S} \in \mathbb{R}^{w \times h}$ of the input sample. The saliency map reflects the sensitivity of the prediction with respect to each spatial location and thus highlights discriminative vein structures.

Specifically, given the gradient feature map $\mathcal{S}$, we select the Top-K spatial locations $\{(x_k, y_k)\}_{k=1}^{K}$ with the highest saliency values. For each selected point, we generate a smooth Gaussian mask centered at (x_k, y_k):

$$\text{Point}(x, y) = \exp\left(-\frac{(x - x_k)^2 + (y - y_k)^2}{\sigma^2}\right), \tag{11}$$

Algorithm 1 StarMixup^{pixel} pseudo-code process.

Require: Beta(α, α) distribution, training dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$, mixing ratio λ, Gaussian function Ga(·) and k is kernel size, `is_vit` denotes the encoder type.

$x \in \mathbb{R}^{w \times h \times c}$
for x_i, y_i in $\mathcal{D}$ loder **do**
  λ = Beta(α, α),
  x_j, y_j = `torch.randperm`(x_i, y_i),
  if $0.3 \leq \lambda \leq 0.7$ **then**
    if `is_vit` **then**
      h = 14, scale = 16, upsample = `nearest`
    end if
    $\mathcal{M}$ = Ga(k, h, σ, scale, upsample),
    $\hat{\lambda}$ according to the Eq. (8),
    $\hat{x} = \mathcal{M} \odot x_i + (1 - \mathcal{M}) \odot x_j$,
    $\hat{y} = \hat{\lambda} * y_i + (1 - \hat{\lambda}) * y_j$.
  else
    $\hat{x} = \lambda * x_i + (1 - \lambda) * x_j$,
    $\hat{y} = \lambda * y_i + (1 - \lambda) * y_j$.
  end if
end for

TABLE II
TPR@FPR (%)↑ OF BASELINE, MIXUP, CUTMIX, SALIENCYMIX, AND STARMIXUP ON TJU600 DATASET.

TJU600	Baseline			MixUp			CutMix			SaliencyMix			StarMixup^{pixel}			StarMixup^{point}		
TPR@FPR	0.01	0.001	0.0001	0.01	0.001	0.0001	0.01	0.001	0.0001	0.01	0.001	0.0001	0.01	0.001	0.0001	0.01	0.001	0.0001
Small-Kernel CNN Models																		
VGG16$_{ICLR'2014}$	92.88	84.75	73.70	99.83	94.86	89.25	96.68	91.16	81.36	96.81	91.18	82.40	98.83	94.88	89.25	98.16	94.78	89.13
ResNet18$_{CVPR'2016}$	98.98	95.88	89.91	99.11	96.58	92.06	99.60	97.45	92.93	99.80	98.38	94.26	99.53	98.05	93.96	99.85	98.26	95.41
ResNet50$_{CVPR'2016}$	97.96	93.58	86.48	99.60	97.65	93.70	99.46	97.36	92.33	99.53	97.28	92.25	99.51	97.96	93.70	99.61	97.61	93.93
ViT-based Global Receptive Field Models																		
ViT-small$_{ICLR'2021}$	–	–	–	95.68	87.15	74.70	89.53	75.28	60.68	90.53	76.41	61.46	94.31	84.75	71.56	94.76	85.20	71.03
ViT-base$_{ICLR'2021}$	–	–	–	96.08	88.55	77.26	90.96	77.18	62.33	90.60	77.83	63.73	94.91	86.15	72.90	94.90	86.35	72.51
DeiT-tiny$_{ICML'2021}$	–	–	–	94.31	85.10	71.36	85.58	70.88	55.26	86.45	71.11	53.66	93.90	83.98	68.70	91.05	79.13	63.28
DeiT-small$_{ICML'2021}$	–	–	–	95.36	87.41	75.58	91.73	79.56	64.83	91.63	79.48	65.71	94.81	87.11	73.65	95.18	85.15	72.46
SwinT-tiny$_{ICCV'2021}$	–	–	–	99.13	96.51	89.81	97.31	87.85	73.65	96.70	88.00	74.15	99.25	96.06	89.65	97.95	91.63	81.86
SwinT-small$_{ICCV'2021}$	–	–	–	99.16	96.25	89.88	96.98	87.60	73.55	97.21	88.88	74.60	99.06	95.96	89.31	98.26	92.38	81.86
SwinT-base$_{ICCV'2021}$	–	–	–	99.35	97.01	91.65	97.76	90.01	77.15	97.31	88.95	76.56	99.23	96.50	90.45	99.23	96.55	90.41
Vein-Specialized Models																		
FVCNN$_{TIFS'2019}$	79.86	65.80	47.90	96.33	90.51	82.65	91.38	79.66	63.51	93.71	84.65	70.63	96.95	91.86	83.35	96.66	90.71	81.60
PVCNN$_{TIFS'2019}$	87.98	74.85	60.58	97.63	94.11	86.73	92.25	83.38	72.48	93.13	84.76	74.33	98.31	93.95	85.90	97.06	92.78	83.00
FVRasNet$_{TIM'2020}$	92.51	79.11	63.16	98.71	95.81	89.05	97.73	94.18	88.36	99.05	96.06	89.51	98.83	95.76	90.11	98.76	95.51	88.93
AMPVNet$_{TIFS'2024}$	93.61	84.66	72.55	98.61	95.76	89.96	96.48	90.16	79.83	96.83	91.23	81.30	98.85	95.68	89.68	98.53	95.66	89.76
RSNet$_{TIFS'2025}$	95.48	86.53	74.63	99.21	96.73	90.95	98.03	85.35	88.70	98.13	94.51	86.56	99.20	96.88	91.95	99.38	96.76	91.65
WTxGRN$_{TIFS'2025}$	95.45	88.53	77.50	99.21	97.11	92.25	99.35	97.43	92.46	99.55	98.31	94.85	99.21	97.21	92.65	99.06	96.96	92.40
Large-Kernel Models (Ours)																		
StarLKNet-tiny	98.18	94.86	86.47	99.78	99.30	96.90	99.81	98.58	95.86	99.66	98.08	94.01	98.65	97.86	92.56	99.88	99.25	97.13
StarLKNet-small	**99.23**	95.96	89.13	**99.88**	**99.71**	**98.53**	**99.91**	**99.05**	**96.81**	**99.83**	**98.66**	**96.28**	99.88	99.65	98.65	**99.88**	**99.58**	**98.60**
StarLKNet-base	98.06	96.45	90.01	99.86	99.63	98.53	99.58	98.65	95.50	99.66	98.63	95.12	**99.90**	**99.71**	**98.65**	97.01	96.65	93.88
StarLKNet-large	98.33	**96.76**	**91.36**	99.45	98.31	94.62	99.45	98.31	94.62	99.68	98.63	95.50	99.88	99.65	98.65	99.06	98.63	98.53
Gains	**+0.25**	**+0.88**	**+1.45**	**+0.05**	**+2.06**	**+4.83**	**+0.31**	**+1.6**	**+3.88**	**+0.03**	**+0.28**	**+1.43**	**+0.37**	**+1.66**	**+4.69**	**+0.03**	**+1.32**	**+3.19**

where σ controls the spatial smoothness and is automatically scaled according to the input resolution. The final StarMask is obtained by aggregating the K Gaussian components:

$$\mathcal{M} = \frac{1}{\max\left(\sum_{k=1}^{K} \text{Point}(x,y)\right)} \sum_{k=1}^{K} \text{Point}(x,y). \tag{12}$$

The mask is normalized to $[0,1]$ and used to perform pixel-wise interpolation as $\hat{x} = \mathcal{M} \odot x_i + (1-\mathcal{M}) \odot x_j$.

Compared to the spatially predefined Gaussian mask in StarMixup^{pixel}, the proposed StarMixup^{point} adaptively focuses on gradient-sensitive regions, which correspond to structurally discriminative vein patterns. This forward-based mechanism enables the mixing process to emphasize informative regions while suppressing background interference, thereby improving robustness and generalization performance.

D. StarLKNet Architecture Specifications

This subsection presents a detailed description of the StarLKNet backbone. As shown in Fig. 4, StarLKNet comprises the following sub-modules. Table III shows the model architectures across kernel sizes, number of layers, and dimensions.

Stem is the first module used as input to the network to capture more details by mapping the raw samples into a higher-dimensional space. Stem comprises three different convolution kernel, a Conv$_{3\times3}$ with step=2, a Conv$_{1\times1}$ with step=1 and a DW Conv$_{3\times3}$ (Depthwise Separable Convolution) with step=2. Batch Normalization (BN) layer and ReLU activation function for reducing the internal covariate shift, stable training, and introducing non-linearity to enhance the representation.

Embedding module is comprised of a Conv$_{1\times1}$ with step=1, two BN layers and a ReLU function. The purpose of the embedding module is to aggregate feature information from the stage inputs during convolution, and to avoid gradient explosion or vanishing.

LaKBlock module comprises 3 convolution kernels: a large DW Conv with k-size, a small DW Conv with s-size, and a normal Conv$_{1\times1}$. A large kernel size aims to extend the *ERF* by capturing more global features, whereas a small kernel size captures more detailed local features. The global and local features are aggregated by a Conv$_{1\times1}$, producing an output that is then fed into the next Channel Gating Neck layer.

Channel Gating Neck module splits the inputs into global channel features z_c and gating features z_g. Firstly, we will further introduce in Section III-E. Since the Neck connects different stages of the architecture, the output of the last DW Conv$_{3\times3}$ must be encoded into a higher-dimensional representation (2×channels).

FC Layer module including an Adaptive Average Pooling and a Linear layer, which is intended to be used for the final classification or other task, where the obtained feature information is subjected to a mapping to obtain the final

Algorithm 2 StarMixup$^{\text{point}}$ pseudo-code process.

Require: Beta(α, α) distribution, training dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$, Gaussian function Ga$(\cdot)$, Top-$K$ points, encoder $f_\theta(\cdot)$, `is_vit` denotes the encoder type.

$x \in \mathbb{R}^{w \times h \times c}$
for x_i, y_i in $\mathcal{D}$ loader **do**
  λ = Beta(α, α)
  x_j, y_j = `torch.randperm`(x_i, y_i)
  $S = \left\| \frac{\partial \mathcal{L}(f_\theta(x_i), y_i)}{\partial x_i} \right\|$
  if `is_vit` **then**
    Downsample S to patch resolution
    scale = 16, upsample = `nearest`
  end if
  $\{(x_k, y_k)\}_{k=1}^{K}$ = Top-K positions from S
  $\text{Point}(x, y) = \exp\left(-\frac{(x-x_k)^2+(y-y_k)^2}{\sigma^2}\right)$
  $\mathcal{M} = \frac{1}{\max(\sum_{k=1}^{K} \text{Point}(x,y))} \sum_{k=1}^{K} \text{Point}(x, y)$
  Normalize $\mathcal{M}$ to $[0, 1]$
  $\hat{x} = \mathcal{M} \odot x_i + (1 - \mathcal{M}) \odot x_j$
  $\hat{y} = \hat{\lambda} y_i + (1 - \hat{\lambda}) y_j$
end for

classification probability distribution.

E. Advanced Module in StarLkNet

This subsection explains two modules in our StarLKNet.

Feature mixing. This paragraph explains the large kernel DW $\text{Conv}_{\text{k}\times\text{k}}$ and small kernel DW $\text{Conv}_{\text{s}\times\text{s}}$ mixing method, aiming to enable the model to capture both local and global features. This allows for stable optimization of the model f_θ.

$$z = \text{ReLU}\big(\text{DW Conv}_{\text{k}\times\text{k}}(z) + \text{DW Conv}_{\text{s}\times\text{s}}(z)\big), \tag{13}$$

where k is set to 3 for a small-size architecture, 5 for a base-size architecture, and 7 for a large-size architecture. In palm vein images, the DW $\text{Conv}_{\text{k}\times\text{k}}$ captures the total distribution of veins and locations. The DW $\text{Conv}_{\text{s}\times\text{s}}$ more captures the thickness of the vein and the direction of rotation.

Gating bottleneck. The bottleneck usually uses small convolution kernels to reduce noise and enhance useful features, which can improve performance and reduce training time, but it can lead to overfitting to local features. The Gating operation can emphasize or suppress features by learning $\text{Conv}_{1\times1}$ weights, combined with an activation function to introduce nonlinearity and complete the feature screening process. In our Channel Gating Neck module, we use the gating to replace the bottleneck, and use two Dilation convolution kernels to retain and further enhance the global features according to Eq. (14):

$$\begin{aligned} \hat{z} =& z + \text{BN}(z_g * z_c) \\ =& z + \text{BN}\big(\text{Gating}(z) * \text{DialConv set}(z)\big), \end{aligned} \tag{14}$$

where the DialConv set$(\cdot)$ denotes the Dilation convolution kernels set with dilation ratio [3, 1], z denotes the input features for the residual connection.

TABLE III
ARCHITECTURE CONFIGURATIONS OF STARLKNET VARIANTS.

Architecture Configurations					
Stage Index	Modules	Tiny	Small	Base	Large
Stage 1	Large Kernel Sizes	27	27	31	31
	Layers	2	2	2	2
	Channels	32	64	128	256
Stage 2	Large Kernel Sizes	27	27	29	19
	Layers	2	2	2	2
	Channels	64	128	256	512
Stage 3	Large Kernel Sizes	27	27	27	27
	Layers	10	18	18	18
	Channels	128	256	512	1024
Stage 4	Large Kernel Sizes	13	13	13	13
	Layers	2	2	2	2
	Channels	256	512	1024	2048
Small Kernel Size		3	3	5	5
Dilation Ratio		[1, 3]			
FC Layer	`Adaptive Average Pooling, Linear`				

IV. EXPERIMENTS

To evaluate the performance of our methods, we conducted experimental comparisons on 4 public palm vein datasets: TJU600 [43], SCUT834 [25], CASIA200 [44], and VERA220 [45]. For each dataset, we compared six cases: baseline (without mixup), MixUp [9], CutMix [10], SaliencyMix [11], StarMixup$^{\text{pixel}}$, and StarMixup$^{\text{point}}$. To ensure a comprehensive comparison, we benchmarked against three categories of models: (i) general-purpose *Small-Kernel CNN Models* (VGG16 [46], ResNet18 [20], and ResNet50); (ii) *Vein-Specialized CNN Models* (FVCNN [12], PVCNN [14], FVRasNet [13], AMPVNet [25], RSNet [26], and WTxGRN [27]); and (iii) *ViT-based Global Receptive Field Models* (ViT [17], DeiT [18], and Swin Transformer (SwinT) [19]).

We evaluate performance using Top-1 accuracy (Acc.), reported as the median value over the final ten training epochs to reflect stable classification; the EER, derived from the false acceptance and false rejection rates at the last epoch; and the True Positive Rate (TPR) at low False Positive Rates (FPR) of 0.01, 0.001, and 0.0001, which measures verification sensitivity under stringent security thresholds. The best results achieved by our methods are highlighted in **bold**, while the best results among the compared methods are underlined. In addition to performance metrics, we also report with model parameters (Param.) and floating-point operations (FLOPs).

A. Datasets

In this subsection, we present details of the four publicly available palm vein datasets. All palm vein images are resized to 224 $\times$ 224 after ROI normalization for fair comparison.

TJU600 [43]: The TJU dataset consists of palm vein images provided for the left and right hands of 300 volunteers. The data from each volunteer were collected 2 times, with about 60 days between the collections. The images were collected from each palm per time point, contains 12,000 images (300 volunteers $\times$ 2 palms $\times$ 10 images $\times$ 2 time periods).

SCUT834 [25]: The SCUT dataset consists of palm vein images provided for the left and right hands of 417 volunteers, captured under unconstrained and weak-cooperative conditions

TABLE IV
TOP-1 ACCURACY(%)↑ AND EER(%)↓ OF BASELINE, MIXUP, CUTMIX, SLIENCYMIX, AND STARMIXUP ON SCUT834, ACC. MEANS ACCURACY.

SCUT834	Param. (M)	FLOPs (G)	Baseline		MixUp		CutMix		SaliencyMix		StarMixup^{pixel}		StarMixup^{point}	
			Acc. ↑	EER ↓	Acc. ↑	EER ↓	Acc. ↑	EER ↓	Acc. ↑	EER ↓	Acc. ↑	EER ↓	Acc. ↑	EER ↓
Small-Kernel CNN Models														
VGG16$_{ICLR'2014}$	137.68	15.5	90.43	1.01	96.98	0.33	92.34	0.92	92.71	0.82	97.51	0.20	96.00	0.46
ResNet18$_{CVPR'2016}$	11.60	1.82	98.01	0.19	99.28	0.09	99.16	0.10	99.06	0.10	99.28	0.08	99.35	0.07
ResNet50$_{CVPR'2016}$	25.22	4.12	97.60	0.20	99.26	0.08	98.85	0.09	98.59	0.10	99.21	0.10	99.35	0.10
ViT-based Global Receptive Field Models														
ViT-small$_{ICLR'2021}$	48.65	9.42	–	–	90.74	1.08	85.42	1.94	83.93	1.97	89.93	1.27	90.17	1.20
ViT-base$_{ICLR'2021}$	86.44	16.86	–	–	91.68	0.93	84.75	2.04	84.58	1.85	90.74	1.20	90.79	1.18
DeiT-tiny$_{ICML'2021}$	5.69	1.08	–	–	88.56	1.30	80.17	2.16	80.46	2.45	89.30	1.37	82.59	2.64
DeiT-small$_{ICML'2021}$	21.99	4.24	–	–	91.08	0.98	85.61	1.85	86.38	1.51	89.06	1.27	82.33	2.16
SwinT-tiny$_{ICCV'2021}$	28.16	4.36	–	–	98.42	0.12	96.71	0.32	96.47	0.43	98.11	0.17	96.76	0.39
SwinT-small$_{ICCV'2021}$	49.48	8.52	–	–	98.23	0.14	96.55	0.37	96.26	0.46	97.99	0.17	96.52	0.43
SwinT-base$_{ICCV'2021}$	87.60	15.14	–	–	98.73	0.10	97.36	0.31	97.27	0.28	98.63	0.14	98.54	0.12
Vein-Specialized CNN Models														
FVCNN$_{TIFS'2019}$	25.00	48.31	89.86	1.45	93.79	0.74	92.73	0.86	92.95	0.85	93.45	0.86	94.65	0.73
PVCNN$_{TIFS'2019}$	4.61	5.45	91.49	1.12	94.12	0.71	92.01	1.01	92.13	0.927	95.52	0.55	96.07	0.60
FVRasNet$_{TIM'2020}$	0.95	0.76	95.47	0.52	97.48	0.26	97.84	0.35	97.96	0.21	97.53	0.30	98.15	0.20
AMPVNet$_{TIFS'2024}$	1.50	0.17	95.49	0.46	97.82	0.36	96.50	0.38	96.43	0.44	97.79	0.29	98.08	0.34
RSNet$_{TIFS'2025}$	2.45	0.18	95.80	0.46	98.51	0.15	97.55	0.33	97.60	0.28	98.39	0.19	98.44	0.19
WTxGRN$_{TIFS'2025}$	17.97	4.50	96.57	0.31	99.35	0.13	99.25	0.07	99.40	0.07	99.30	0.15	99.47	0.10
Large-Kernel Models (Ours)														
StarLKNet-tiny	3.21	1.12	98.03	0.26	99.38	0.12	99.04	0.17	99.16	0.15	99.42	0.15	99.54	0.17
StarLKNet-small	12.62	4.10	97.79	0.31	99.62	0.12	99.47	**0.07**	99.28	0.09	99.59	0.09	99.52	0.13
StarLKNet-base	42.30	14.42	**98.30**	**0.19**	99.62	0.10	**99.54**	0.08	99.50	0.11	**99.74**	0.07	99.66	**0.06**
StarLKNet-large	151.08	52.42	98.15	0.24	**99.76**	**0.07**	99.52	0.10	**99.57**	**0.06**	99.69	**0.05**	**99.81**	0.09
Gains	–	–	**+0.29**	**-0.00**	**+0.41**	**-0.01**	**+0.27**	**-0.00**	**+0.17**	**-0.01**	**+0.44**	**-0.03**	**+0.34**	**-0.01**

over three years. As approximately 10 vein images were collected from each palm, the dataset contains 8340 near-infrared images (417 volunteers × two palms × 10 images).

VERA220 [45]: The VERA dataset contains palm vein images of the left and right hands of 110 volunteers, acquired over two periods. As five palm vein images were collected for each hand at each time, the dataset contains 2200 images (110 volunteers × 2 palms × 5 images × 2 time periods).

CASIA200 [44]: The CASIA palm vein dataset contains palm vein images of the left and right hands of 100 volunteers, acquired over two sessions with more than one month between them. As 3 vein images were collected from each palm per session, the dataset contains 1,200 images (100 volunteers × 2 palms × 3 images × 2 time periods).

B. Implementations

For all the palm vein datasets, we use RandomFlip and 3-pixel padding with RandomCrop as the basic augmentation methods. The batch size for the experiments was set to 32, and training was carried out for 600 epochs. For the advanced CNN models, including ResNet18, ResNet50, FVRasNet, and our proposed StarLKNet, the learning rate was initialized to 0.1. For VGG16, the learning rate was set to 0.01 and adjusted using a cosine scheduler. A momentum of 0.9 and a weight decay of 0.0001 were used with the SGD optimizer. For the FVCNN, PVCNN, RSNet, and WTxGRN, the AdamW [47] optimizer with a learning rate of 0.001 and a weight decay of 0.01 was used. For ViT-based models, e.g., ViT (small, base), DeiT (tiny, small), and SwinT (tiny, small, and base), we follow the settings of OpenMixup [48]; the learning rate was set to 0.0001, adjusted by a cosine scheduler, and warmup under 150 epochs with a 0.0001 ratio. Optimized by AdamW optimizer with a 0.001 weight decay. In all ViT-based models, we used label smoothing with ϵ=0.1.

C. Overall Recognition Performance

Table I, Table VI, Table IV, and Table VIII show the performance of our methods in terms of 2 metrics: Acc. and EER. Table I shows that StarLKNet models consistently outperform all competitors under every augmentation, achieving the highest Acc. (98.78% with StarMixup^{point}) and lowest EER (0.16% with MixUp) on TJU600. Compared to the best existing models, our model achieves up to 4.02% higher Acc. and reduces EER by up to 0.44%, while using only 12.56M parameters and 4.12G FLOPs. Moreover, our proposed StarMixup^{pixel} and StarMixup^{point} consistently boost performance over baseline and other mixup methods, often yielding the best results for our models. These results validate the effectiveness of our large-kernel design and augmentation strategy. To further validate the performance, we conducted experiments on three additional

TABLE V
TPR@FPR(%)↑ OF BASELINE, MIXUP, CUTMIX, SALIENCYMIX, AND STARMIXUP ON SCUT834.

SCUT834	Baseline			MixUp			CutMix			SaliencyMix			StarMixup^{pixel}			StarMixup^{point}		
TPR@FPR	0.01	0.001	0.0001	0.01	0.001	0.0001	0.01	0.001	0.0001	0.01	0.001	0.0001	0.01	0.001	0.0001	0.01	0.001	0.0001
Small-Kernel CNN Models																		
VGG16$_{ICLR'2014}$	98.99	96.01	90.19	99.80	99.23	97.72	99.25	97.41	92.61	99.25	97.41	92.82	99.88	99.25	96.28	99.73	98.75	96.28
ResNet18$_{CVPR'2016}$	99.92	99.73	98.99	99.97	99.90	99.68	99.97	99.88	99.61	99.97	99.92	99.64	99.95	99.95	99.81	99.97	99.92	99.64
ResNet50$_{CVPR'2016}$	99.95	99.49	98.58	99.97	99.90	99.68	99.97	99.90	99.30	99.97	99.85	99.16	99.97	99.88	99.76	99.97	99.90	99.68
ViT-based Global Receptive Field Models																		
ViT-small$_{ICLR'2021}$	–	–	–	98.82	85.22	89.49	97.26	91.41	81.51	97.07	90.11	78.94	98.56	94.91	88.75	98.48	94.75	88.96
ViT-base$_{ICLR'2021}$	–	–	–	99.18	96.30	91.24	96.83	90.67	80.00	97.26	91.43	81.15	98.68	95.68	89.08	98.65	95.68	89.42
DeiT-tiny$_{ICML'2021}$	–	–	–	98.36	94.48	87.14	96.40	88.03	74.50	96.01	87.67	75.58	98.22	94.60	87.84	95.89	89.20	78.01
DeiT-small$_{ICML'2021}$	–	–	–	99.08	95.77	90.52	97.33	91.84	81.53	97.81	92.42	83.31	98.56	94.58	87.67	96.74	89.59	78.03
SwinT-tiny$_{ICCV'2021}$	–	–	–	99.92	99.83	99.13	99.88	99.20	97.12	99.78	98.99	96.76	99.92	99.73	98.82	99.92	99.30	97.26
SwinT-small$_{ICCV'2021}$	–	–	–	99.97	99.73	98.99	99.92	99.11	96.64	99.78	98.94	96.30	99.97	99.66	98.65	99.78	99.11	96.47
SwinT-base$_{ICCV'2021}$	–	–	–	99.97	99.90	99.40	99.95	99.40	97.43	99.95	99.20	97.48	99.97	99.83	99.16	99.95	99.76	99.13
Vein-Specialized Models																		
FVCNN$_{TIFS'2019}$	98.24	95.15	89.52	99.35	97.72	93.57	99.18	97.41	93.33	99.23	97.29	93.50	99.18	97.55	93.35	99.44	98.15	94.79
PVCNN$_{TIFS'2019}$	98.87	96.54	91.46	99.35	97.72	94.34	98.99	96.54	91.96	99.25	96.29	91.96	99.66	98.56	95.56	99.64	98.39	96.45
FVRasNet$_{TIM'2020}$	99.68	98.70	96.59	99.92	99.40	97.91	99.80	99.37	98.39	99.90	99.61	98.87	99.88	99.35	98.11	99.92	99.61	98.66
AMPVNet$_{TIFS'2024}$	99.71	98.77	96.49	99.90	99.35	98.41	99.83	98.99	97.43	99.78	98.94	97.55	99.88	99.44	98.48	99.88	99.32	98.46
RSNet$_{TIFS'2025}$	99.71	98.96	97.00	99.92	99.76	98.99	99.83	99.28	98.15	99.92	99.47	98.46	99.97	99.71	98.92	99.95	99.64	98.99
WTxGRN$_{TIFS'2025}$	99.88	99.30	97.57	99.92	99.83	99.61	99.97	99.95	99.76	99.97	99.97	99.97	99.97	99.85	99.66	99.95	99.90	99.73
Large-Kernel Models (Ours)																		
StarLKNet-tiny	99.88	99.52	98.60	99.95	99.88	99.59	99.97	99.80	99.52	99.97	99.73	99.42	99.95	99.83	99.52	99.95	99.85	99.68
StarLKNet-small	**99.92**	99.49	98.48	99.95	99.88	**99.78**	**99.97**	**99.95**	99.76	99.95	99.90	99.73	99.95	99.92	99.71	99.97	99.88	99.71
StarLKNet-base	99.92	99.61	**99.04**	99.97	99.92	99.78	99.97	99.92	**99.85**	**99.97**	99.88	99.78	99.97	99.92	**99.83**	**99.97**	**99.92**	99.83
StarLKNet-large	99.85	**99.71**	98.94	**99.97**	**99.92**	99.78	99.97	99.90	99.83	99.97	**99.95**	**99.80**	**99.97**	**99.95**	99.80	99.97	99.92	**99.85**
Gains	**-0.03**	**-0.02**	**+0.05**	**+0.00**	**+0.02**	**+0.10**	**+0.00**	**+0.05**	**+0.09**	**+0.00**	**-0.02**	**-0.17**	**+0.00**	**+0.00**	**+0.02**	**+0.00**	**+0.00**	**0.12**

datasets: SCUT834, VERA220, and CASIA200. On SCUT834 (Table IV), our StarLKNet models consistently outperform all competitors, with StarLKNet-large achieving the highest Acc. (99.81% with StarMixup^{point}) and the lowest EER (0.05% with StarMixup^{pixel}). On VERA220 (Table VI) and CASIA200 (Table VIII), our models also exhibit substantial gains.

D. Performance under Low FPR

In high-security biometric applications, the ability to maintain a high TPR at extremely low FPR is critical, as it directly reflects the reliability of the model under stringent authentication thresholds. While overall accuracy and EER provide a general performance overview, they do not fully capture verification sensitivity in the low FPR. Therefore, we further evaluate TPR at FPRs of 0.01, 0.001, and 0.0001 on four datasets, and present the Receiver Operating Characteristic (ROC) curves to comprehensively illustrate the trade-off between TPR and FPR over the full range of operating points.

As shown in Table II, StarLKNet models consistently achieve the highest TPR across all FPR thresholds, particularly when combined with StarMixup. For example, StarLKNet-small achieves TPRs of 99.88%, 99.71%, and 98.53% at different FPRs, respectively, outperforming the best competing method by up to 4.83% at the most stringent FPR of 0.0001. Similar improvements are observed under augmentations, e.g., with StarMixup^{pixel}, StarLKNet-base achieves TPR of 99.90%, 99.71%, and 98.65%, surpassing all prior models. Gains are particularly pronounced at the lowest FPR, demonstrating the superior discriminative power and robustness of our approach. Consistent trends are also observed on the SCUT834 (Table V), VERA220 (Table VII), and CASIA200 (Table VIII) datasets, where our methods maintain leading TPR across all FPR levels.

In addition to the tabulated results, we present the ROC curves for various models under different augmentation strategies on three datasets (Fig. 6). The curves further corroborate the quantitative findings: StarLKNet-small and StarLKNet-base consistently achieve the highest TPR at low FPR, underscoring their strong verification capability. Moreover, StarMixup^{pixel} consistently boosts performance, outperforming both MixUp and CutMix across all datasets. These results collectively validate the effectiveness of our large-kernel architecture and the proposed augmentation in enhancing palm vein verification under stringent security requirements.

E. Robustness Experiments

To demonstrate the robustness of StarLKNet, we conducted 2 scenarios for the experiments. (a). Occlusion classification [49] and (b). t-SNE clustering [50].

Occlusion classification: we randomly occluded with 16 $\times$ 16 size patches, with the occlusion ratio gradually increasing from 0% to 10%. Then, input the occluded test sets into different models for classification. Since vein images contain

TABLE VI
TOP-1 ACCURACY (%)↑ AND EER (%)↓ OF BASELINE, MIXUP, CUTMIX, SALIENCYMIX, AND STARMIXUP ON VERA220, ACC. MEANS ACCURACY.

VERA220	Param. (M)	FLOPs (G)	Baseline		MixUp		CutMix		SaliencyMix		StarMixup^{pixel}		StarMixup^{point}	
			Acc. ↑	EER ↓	Acc. ↑	EER ↓	Acc. ↑	EER ↓	Acc. ↑	EER ↓	Acc. ↑	EER ↓	Acc. ↑	EER ↓
Small-Kernel CNN Models														
VGG16$_{ICLR'2014}$	135.16	15.50	66.91	8.10	95.09	1.26	70.55	8.20	69.64	8.97	93.82	1.54	85.55	3.01
ResNet18$_{CVPR'2016}$	11.29	1.82	71.91	4.53	95.55	1.26	80.18	3.99	82.82	3.01	96.55	1.21	95.91	0.84
ResNet50$_{CVPR'2016}$	23.96	4.12	66.91	7.00	93.91	1.64	77.09	3.73	71.73	5.28	96.00	0.88	89.00	1.99
Vein-Specialized Models														
FVCNN$_{TIFS'2019}$	24.69	48.31	69.36	5.39	90.27	2.34	78.64	3.34	82.64	2.79	89.27	2.88	89.18	1.86
PVCNN$_{TIFS'2019}$	4.29	5.45	54.73	12.40	82.18	2.66	52.64	12.99	60.45	11.36	83.45	2.62	73.18	6.12
FVRasNet$_{TIM'2020}$	0.79	0.76	67.27	6.47	90.73	1.64	69.91	6.21	72.91	5.19	88.73	1.84	83.91	2.44
AMPVNet$_{TIFS'2024}$	1.18	0.17	75.00	4.10	95.64	0.93	83.09	2.72	84.73	2.28	96.18	0.89	93.64	0.89
RSNet$_{TIFS'2025}$	2.14	0.18	70.73	5.56	94.73	1.02	78.00	4.97	79.00	4.55	95.55	0.91	91.45	1.47
WTxGRN$_{TIFS'2025}$	17.50	4.50	69.45	4.99	95.18	0.84	86.91	3.72	86.73	2.84	95.55	0.91	91.45	1.47
Large-Kernel Models (Ours)														
StarLKNet-tiny	3.08	1.12	83.55	2.91	98.00	0.72	86.70	3.02	85.55	3.01	97.73	**0.52**	**98.82**	0.54
StarLKNet-small	12.37	4.12	84.64	**2.31**	**98.55**	0.65	88.18	2.46	**89.72**	2.45	**98.45**	0.55	98.64	0.71
StarLKNet-base	41.67	14.42	**87.45**	2.39	98.00	**0.41**	89.73	2.34	89.27	3.05	98.00	0.99	98.64	**0.50**
StarLKNet-large	149.82	52.42	83.45	3.38	94.27	2.04	**90.91**	**1.95**	89.18	**1.99**	94.18	2.79	97.73	0.79
Gains	–	–	**+12.45**	**-1.79**	**+3.00**	**-0.43**	**+4.00**	**-0.77**	**+2.99**	**-0.29**	**+1.90**	**-0.36**	**+2.91**	**-0.34**

TABLE VII
TPR@FPR (%)↑ OF BASELINE, MIXUP, CUTMIX, SALIENCYMIX, AND STARMIXUP ON VERA220 DATASET.

VERA220	Baseline			MixUp			CutMix			SaliencyMix			StarMixup^{pixel}			StarMixup^{point}		
TPR@FPR	0.01	0.001	0.001	0.01	0.001	0.001	0.01	0.001	0.001	0.01	0.001	0.001	0.01	0.001	0.001	0.01	0.001	0.001
Small-Kernel CNN Models																		
VGG16$_{ICLR'2014}$	80.36	64.54	43.54	98.72	96.51	91.54	80.27	68.81	51.90	81.18	66.63	49.72	98.36	95.54	85.18	95.00	87.09	69.64
ResNet18$_{CVPR'2016}$	88.91	70.63	46.36	98.81	97.81	92.63	90.81	80.27	68.09	93.00	83.54	72.91	98.81	97.45	95.00	99.09	97.09	91.54
ResNet50$_{CVPR'2016}$	81.54	64.00	41.90	98.09	96.00	91.00	90.00	76.45	62.90	85.64	70.72	54.09	99.00	97.27	94.09	96.72	92.54	80.27
Vein-Specialized Models																		
FVCNN$_{TIFS'2019}$	85.81	67.36	41.90	96.54	92.36	80.82	93.81	79.09	53.54	93.81	80.91	62.36	96.36	91.54	82.09	97.45	91.45	78.27
PVCNN$_{TIFS'2019}$	67.18	48.54	25.72	95.18	84.09	62.09	66.54	47.09	28.63	71.81	55.54	36.36	95.81	87.27	65.54	85.81	70.45	43.27
FVRasNet$_{TIM'2020}$	82.27	65.90	41.99	97.72	92.27	78.81	83.36	68.72	53.18	86.18	73.27	57.63	97.54	89.82	78.27	95.00	86.00	77.18
AMPVNet$_{TIFS'2024}$	92.36	75.36	51.63	99.09	97.81	93.63	95.45	85.27	73.81	96.00	87.00	73.72	99.27	97.93	94.00	99.18	96.09	89.36
RSNet$_{TIFS'2025}$	85.81	69.45	50.54	98.90	97.18	91.81	90.81	77.36	59.54	90.18	78.72	62.81	99.45	97.73	92.09	98.64	96.00	91.18
WTxGRN$_{TIFS'2025}$	85.90	67.09	47.90	99.36	97.90	92.45	93.54	88.27	80.72	94.36	87.63	79.90	99.18	97.27	92.81	98.09	94.36	83.18
Large-Kernel Models (Ours)																		
StarLKNet-tiny	94.64	83.73	66.82	**99.55**	98.73	96.64	93.55	87.73	79.82	94.36	86.27	74.18	99.36	98.91	96.00	99.36	98.91	97.45
StarLKNet-small	**96.18**	84.45	67.36	99.45	**99.18**	**97.64**	95.55	89.55	79.00	95.72	90.18	**79.73**	**99.64**	**98.91**	96.45	99.45	**99.09**	97.18
StarLKNet-base	95.36	**88.45**	**72.09**	99.45	98.64	97.09	96.45	91.45	80.91	95.73	**91.36**	78.55	99.09	98.82	**96.91**	**99.72**	99.00	**98.09**
StarLKNet-large	94.27	83.09	66.18	97.73	96.36	86.82	**97.00**	**92.54**	**86.00**	**96.18**	91.00	78.82	96.91	96.36	66.27	99.36	98.64	97.64
Gains	**+3.82**	**+13.09**	**+20.46**	**+0.19**	**+1.28**	**+4.01**	**+1.55**	**+4.27**	**+5.28**	**+0.18**	**+3.73**	**-0.17**	**+0.19**	**+0.98**	**+1.91**	**+0.54**	**+2.00**	**+6.55**

both continuous and discrete features, a mask patch may interrupt these features when it falls in a region with constant veins. This challenges the model and provides an additional configuration to assess its robustness to vein features. From Fig. 7, we observe that StarLKNet-small achieves the highest accuracy for occlusion ratios from 0% to 10%, which shows that the large kernels are beneficial for robustness.

t-SNE clustering: We compare the trained models by visualizing the representation of our StarLKNet and ResNet18 in Fig. 8. A close look at t-SNE shows that images of the same class cluster together, indicating better learning. Notably, StarLKNet exhibits distinct, more cohesive clusters with well-defined margins between classes, suggesting that the network consistently learns class-specific discriminative features.

F. Ablation Study

Our ablation experiments aim to evaluate the effectiveness of the StarMixup augmentation method, the Gating module, and the Large Kernel module, and to conduct hyperparameter sensitivity analysis in our approach.

Augmentation ablation: We perform experiments using ResNet18, FVRasNet, RSNet and StarLKNet-tiny on

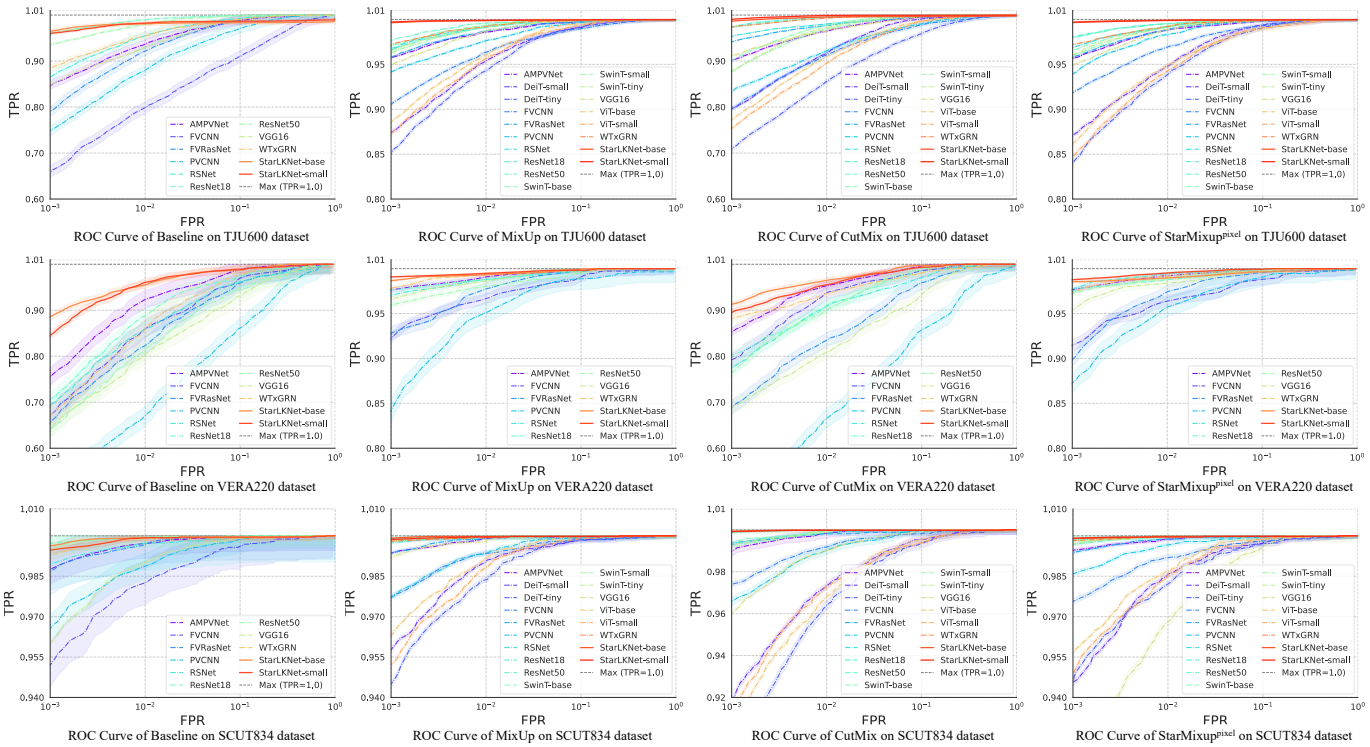


Fig. 6. ROC curves of different models under various data augmentation strategies on three datasets. From left to right: Baseline (no augmentation), MixUp, CutMix, and our proposed StarMixup$^{\text{pixel}}$. ***Top row***: ROC curves on the TJU600 dataset. ***Middle row***: ROC curves on the VERA220 dataset. ***Bottom row***: ROC curves on the SCUT834 dataset. Across all subplots, the x-axis represents the FPR on a log scale, and the y-axis represents the TPR.

TABLE VIII
TOP-1 ACCURACY(%)↑ AND EER(%)↓ RESULTS ON CASIA200 DATASET.

CASIA200	Param. (M)	FLOPs (G)	Accuracy (%) ↑	EER (%) ↓	TPR@FPR			
					0.1	0.01	0.001	0.001
Small-Kernel CNN Models								
VGG16$_{\text{ICLR'2014}}$	135.08	15.5	40.17	15.33	81.50	55.83	23.83	6.33
ResNet18$_{\text{CVPR'2016}}$	11.28	1.82	74.00	4.35	**97.50**	87.83	69.33	40.17
ResNet50$_{\text{CVPR'2016}}$	23.92	4.12	45.83	16.71	79.33	60.67	24.17	3.00
Vein-Specialized CNN Models								
FVCNN$_{\text{TIFS'2019}}$	24.68	48.31	68.00	7.51	93.00	81.33	61.50	27.33
PVCNN$_{\text{TIFS'2019}}$	4.28	5.45	55.67	15.00	81.00	66.17	35.00	7.17
FVRasNet$_{\text{TIM'2020}}$	0.79	0.76	52.33	18.59	79.17	66.33	31.00	0.0
AMPVNet$_{\text{TIFS'2024}}$	1.17	0.17	69.83	6.66	96.00	84.00	65.67	44.67
RSNet$_{\text{TIFS'2025}}$	2.13	0.18	70.00	6.16	95.17	85.50	62.83	35.50
WTxGRN$_{\text{TIFS'2025}}$	17.48	4.5	72.50	5.89	96.00	87.50	65.17	36.67
Large-Kernel Models (Ours)								
StarLKNet-base	41.65	14.42	**76.50**	**4.32**	97.00	**89.33**	**71.00**	**51.00**
Gains	–	–	**+2.50**	**-0.03**	**-0.50**	**+1.50**	**+1.67**	**+6.33**

the TJU600 dataset. The aim is to verify the effect of StarMixup$^{\text{pixel/point}}$ and the [0.3,0.7] threshold setting. Table IX demonstrates that applying StarMixup$^{\text{pixel}}$ is detrimental to the model in certain instances, resulting in a 0.44% reduction in Top-1 accuracy compared to the SaliencyMix. As illustrated in Fig. 5, the mask is observed to concentrate on the central region when λ is less than 0.3 and on the periphery when it is greater than 0.7. However, when λ is between 0.3 and 0.7, the mask achieves an additional boost of 0.51%, 0.33%, 0.20%,

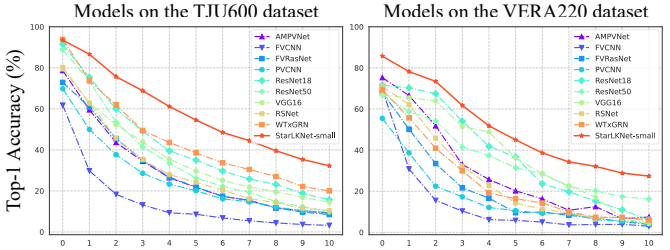


Fig. 7. Occlusion classification curves of different models without data augmentation. ***Left:*** Top-1 accuracy on the TJU600 dataset under test-set random patch masking. ***Right:*** Top-1 accuracy on the VERA220 dataset under the same occlusion protocol, validating cross-dataset generalization.

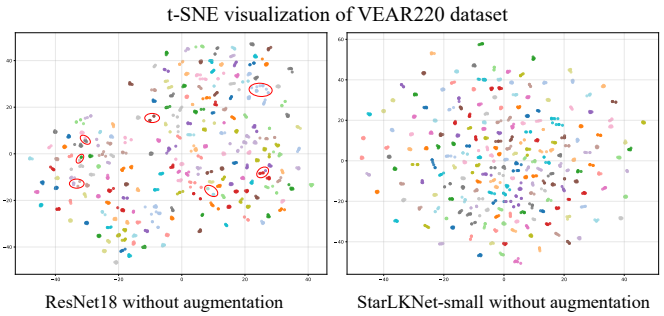


Fig. 8. t-SNE visualization of ResNet18 and StarLKNet-small on the VERA220 dataset. We plot some incorrect clustering using red circles. It is seen that StarLKNet-small can extract more robust features in the hidden space.

and 0.26% on ResNet18, FVRasNet, RSNet, and StarLKNet-tiny, respectively. Meanwhile, the StarMixup$^{\text{point}}$ could improve the gains of 1.25% and 1.10% in some models.

Module ablation: We performed module ablation studies on

TABLE IX
ABLATION EXPERIMENTS ABOUT STARMIXUP^{PIXEL} OR STARMIXUP^{POINT} AND THRESHOLD SETTING METHOD ON THE TJU600 BASED RESNET18, FVRASNET, RSNET, AND STARLKNET-TINY.

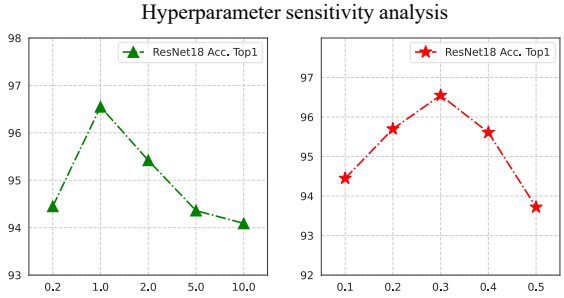
VERA220	ResNet18	FVRasNet	RSNet	StarLKNet-tiny
Baseline	91.32	72.88	80.48	92.42
Other mixup augmentations				
MixUp [9]	94.03	92.08	93.13	95.73
CutMix [10]	94.32	91.77	92.77	96.75
SaliencyMix [11]	95.15	92.03	90.20	96.18
Our augmentation				
StarMixup^{pixel}	93.55	90.27	92.73	94.31
w. smooth mask	94.71	92.65	93.53	96.32
w. threshold	95.22	**92.98**	**93.73**	96.58
StarMixup^{point}	**96.47**	91.97	93.67	**97.68**

TABLE X
ABLATION EXPERIMENTS ABOUT STARLKNET-SMALL BASELINE AND THEIR MODULES EFFECTIVE ON THE VERA220 AND TJU600 DATASETS.

Modules	TJU600				VERA220			
	Param.	FLOPs	Acc.	EER	Param.	FLOPs	Acc.	EER
Vanilla small-kernel CNN with kernel size of 3								
ResNet18	11.48	1.82	91.32	1.03	11.29	1.82	71.91	4.53
With large kernel Conv$_{k\times k}$								
k=7	4.12	0.79	92.41	0.87	3.92	0.79	79.13	3.21
k=14	4.76	0.97	91.45	**1.01**	4.56	0.97	83.63	2.63
k=27	7.68	1.82	89.15	1.95	7.48	1.82	72.18	4.82
Other modules								
+ DialConv set	12.32	4.03	90.77	1.65	12.13	4.03	78.45	4.70
+ Gating	12.56	4.12	**93.58**	1.53	12.37	4.12	**84.64**	**2.31**

TJU600 and VERA220 to validate the key components of StarLKNet (Table X). Compared to ResNet18, simply increasing the kernel size to 7 improves accuracy, but further increasing to 14 or 27 degrades performance, indicating that naive kernel scaling alone is suboptimal. Adding the *DialConv set* reduces accuracy (from 91.32% to 90.77% on TJU600), suggesting that uncontrolled receptive field expansion introduces noise. However, when combined with the *Gating* module, the full model achieves the best results: 93.58% accuracy and 1.53% EER on TJU600, and 84.64% accuracy with 2.31% EER on VERA220. This demonstrates the synergistic effect of DialConv set and gating, where gating effectively filters the high-dimensional features expanded by dilated convolutions, retaining only the most discriminative information.

Hyperparameter analysis: Fig. 9 shows two hyperparameter of StarMixup^{pixel}, i.e., the α of Beta distribution and threshold ratio. The distinct values of α give rise to different distributions of the Beta. A smaller α makes it more likely to sample extreme values near 0 or 1, while α equal to 1 represents a uniform distribution. A larger α, on the other hand, facilitates sampling values close to 0.5. We evaluated five values and found the optimal value for StarMixup^{pixel}, which is 1. The threshold ratio range corresponds to the degree of the mask generated by StarMixup^{pixel}. In Section III-C, we visualized the masks across



(a). α ratio (b). threshold ratio

Fig. 9. Hyperparameter sensitivity analysis of Top-1 accuracy on the VERA220 dataset with StarMixup^{pixel}. (a) Top-1 accuracy across different α ratios, peaking at ratio = 1.0. (b) Top-1 accuracy across different threshold ratios, with optimal performance at ratio = 0.3.

different ratios and conducted a sensitivity analysis. Finally, we found that the range [0.3, 0.7] is optimal for StarMixup^{pixel}.

V. CONCLUSION

In this paper, we propose StarLKNet, a CNN-based palm vein recognition network with large kernels that incorporates the StarMixup data augmentation method and the StarLKNet architecture. StarMixup generates smooth masks for image interpolation via a Gaussian function, thereby alleviating overfitting caused by limited training data and enhancing the model robustness. The large kernel-based network captures more globally effective features through the large *ERF* and the screening capability of the gating mechanism, thereby stabilizing the recognition capability of classier. Extensive recognition experiments and analysis studies are conducted to verify the outstanding performance of the proposed method.

REFERENCES

[1] Yu Liu, Fangyin Wei, Jing Shao, Lu Sheng, Junjie Yan, and Xiaogang Wang. Exploring disentangled feature representation beyond face identification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2080–2089, 2018.
[2] Tarang Chugh, Kai Cao, and Anil K Jain. Fingerprint spoof buster: Use of minutiae-centered patches. *IEEE Transactions on Information Forensics and Security*, 13(9):2190–2202, 2018.
[3] Yiquan Wu, Hongchao Liao, Hongyu Zhu, Xin Jin, Shuqiang Yang, and Huafeng Qin. Adversarial contrastive learning based on image generation for palm vein recognition. In *2023 2nd International Conference on Artificial Intelligence and Intelligent Information Processing (AIIIP)*, pages 18–24. IEEE, 2023.
[4] Chao Fan, Junhao Liang, Chuanfu Shen, Saihui Hou, Yongzhen Huang, and Shiqi Yu. Opengait: Revisiting gait recognition towards better practicality. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9707–9716, 2023.
[5] Huafeng Qin, Hongyu Zhu, Xin Jin, Qun Song, Mounim A El-Yacoubi, and Xinbo Gao. Emmixformer: Mix transformer for eye movement recognition. *IEEE Transactions on Instrumentation and Measurement*, 2025.
[6] Shuqiang Yang, Yiquan Wu, Xin Jin, Mounim El Yacoubi, and Huafeng Qin. Cgan-da: A cross-modality domain adaptation model for hand-vein biometric-based authentication. *JOURNAL OF Cyber-Physical-Social Intelligence*, 1:3–12, 2022.
[7] Wenxiong Kang, Yuting Lu, Dejian Li, and Wei Jia. From noise to feature: Exploiting intensity distribution as a novel soft biometric trait for finger vein recognition. *IEEE Transactions on Information Forensics and Security*, 14:858–869, 2019.
[8] Guoqing Wang, Changming Sun, and Arcot Sowmya. Multi-weighted co-occurrence descriptor encoding for vein recognition. *IEEE Transactions on Information Forensics and Security*, 15:375–390, 2020.

[9] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

[10] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *International Conference on Computer Vision (ICCV)*, pages 6023–6032, 2019.

[11] AFM Shahab Uddin, Mst Sirazam Monira, Wheemyung Shin, TaeChoong Chung, and Sung-Ho Bae. Saliencymix: A saliency guided data augmentation strategy for better regularization. In *International Conference on Learning Representations*, 2020.

[12] Rig Das, Emanuela Piciucco, Emanuele Maiorana, and Patrizio Campisi. Convolutional neural network for finger-vein-based biometric identification. *IEEE Transactions on Information Forensics and Security*, 14(2):360–373, 2018.

[13] Weili Yang, Wei Luo, Wenxiong Kang, Zhixing Huang, and Qiuxia Wu. Fvras-net: An embedded finger-vein recognition and antispoofing system using a unified cnn. *IEEE Transactions on Instrumentation and Measurement*, 69(11):8690–8701, 2020.

[14] Huafeng Qin, Mounim A El-Yacoubi, Yantao Li, and Chongwen Liu. Multi-scale and multi-direction gan for cnn-based single palm-vein identification. *IEEE Transactions on Information Forensics and Security*, 16:2652–2666, 2021.

[15] Xiaohan Ding, Xiangyu Zhang, Jungong Han, and Guiguang Ding. Scaling up your kernels to 31x31: Revisiting large kernel design in cnns. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11963–11975, 2022.

[16] Shiwei Liu, Tianlong Chen, Xiaohan Chen, Xuxi Chen, Qiao Xiao, Boqian Wu, Tommi Kärkkäinen, Mykola Pechenizkiy, Decebal Mocanu, and Zhangyang Wang. More convnets in the 2020s: Scaling up kernels beyond 51x51 using sparsity. *arXiv preprint arXiv:2207.03620*, 2022.

[17] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021.

[18] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, 2021.

[19] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *International Conference on Computer Vision (ICCV)*, 2021.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[21] Xin Jin, Hongyu Zhu, Simon Fong, João Alexandre Lobo Marques, Huafeng Qin, and Yun Jiang. Starmixup: A more suitable mixup method for palm-vein identification. In *2025 7th International Symposium on Computational and Business Intelligence (ISCBI)*, pages 83–87. IEEE, 2025.

[22] Naoto Miura, Akio Nagasaka, and Takafumi Miyatake. Extraction of finger-vein patterns using maximum curvature points in image profiles. *IEICE Trans. Inf. Syst.*, 90-D:1185–1194, 2007.

[23] Beining Huang, Yanggang Dai, Rongfeng Li, Darun Tang, and Wenxin Li. Finger-vein authentication based on wide line detector and pattern normalization. In *2010 20th international conference on pattern recognition*, pages 1269–1272. IEEE, 2010.

[24] Jiaquan Shen, Ningzhong Liu, Chenglu Xu, Han Sun, Yushun Xiao, Deguang Li, and Yongxin Zhang. Finger vein recognition algorithm based on lightweight deep convolutional neural network. *IEEE Transactions on Instrumentation and Measurement*, 71:1–13, 2022.

[25] Dacan Luo, Yitao Qiao, Di Xie, Shifeng Zhang, and Wenxiong Kang. Palm vein recognition under unconstrained and weak-cooperative conditions. *IEEE Transactions on Information Forensics and Security*, 19:4601–4614, 2024.

[26] Dacan Luo, Junduan Huang, Weili Yang, M Saad Shakeel, and Wenxiong Kang. Rsnet: Region-specific network for contactless palm vein authentication. *IEEE Transactions on Information Forensics and Security*, 2025.

[27] Huafeng Qin, Yuming Fu, Jing Chen, Qun Song, Yantao Li, Mounim A El-Yacoubi, and Dexing Zhong. Wtxgrn: Wavelet transform-based extended gated recurrent network for palm vein recognition. *IEEE Transactions on Information Forensics and Security*, 2025.

[28] Jaejun Yoo, Namhyuk Ahn, and Kyung-Ah Sohn. Rethinking data augmentation for image super-resolution: A comprehensive analysis and a new strategy. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8375–8384, 2020.

[29] Viktor Olsson, Wilhelm Tranheden, Juliano Pinto, and Lennart Svensson. Classmix: Segmentation-based data augmentation for semi-supervised learning. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 1369–1378, 2021.

[30] Huaxiu Yao, Yiping Wang, Linjun Zhang, James Y Zou, and Chelsea Finn. C-mixup: Improving generalization in regression. *Advances in neural information processing systems*, 35:3361–3376, 2022.

[31] Hsin-Ping Chou, Shih-Chieh Chang, Jia-Yu Pan, Wei Wei, and Da-Cheng Juan. Remix: rebalanced mixup. In *Computer Vision–ECCV 2020 Workshops: Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16*, pages 95–110. Springer, 2020.

[32] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *International Conference on Machine Learning*, pages 6438–6447, 2019.

[33] Jie Qin, Jiemin Fang, Qian Zhang, Wenyu Liu, Xingang Wang, and Xinggang Wang. Resizemix: Mixing data with preserved object information and true labels. *arXiv preprint arXiv:2012.11101*, 2020.

[34] Siyuan Li, Zicheng Liu, Zedong Wang, Di Wu, Zihan Liu, and Stan Z. Li. Boosting discriminative visual representation learning with scenario-agnostic mixup. *ArXiv*, abs/2111.15454, 2021.

[35] Zicheng Liu, Siyuan Li, Ge Wang, Lirong Wu, Cheng Tan, and Stan Z Li. Harnessing hard mixed samples with decoupled regularizer. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[36] Jang-Hyun Kim, Wonho Choo, and Hyun Oh Song. Puzzle mix: Exploiting saliency and local statistics for optimal mixup. In *International Conference on Machine Learning*, pages 5275–5285. PMLR, 2020.

[37] JangHyun Kim, Wonho Choo, Hosan Jeong, and Hyun Oh Song. Co-mixup: Saliency guided joint mixup with supermodular diversity. In *International Conference on Learning Representations*, 2020.

[38] Zicheng Liu, Siyuan Li, Di Wu, Zihan Liu, Zhiyuan Chen, Lirong Wu, and Stan Z Li. Automix: Unveiling the power of mixup for stronger classifiers. In *European Conference on Computer Vision*, pages 441–458. Springer, 2022.

[39] Huafeng Qin, Xin Jin, Yun Jiang, Mounîm El-Yacoubi, and Xinbo Gao. Adversarial automixup. In *The Twelfth International Conference on Learning Representations*, 2024.

[40] Khawar Islam, Muhammad Zaigham Zaheer, Arif Mahmood, and Karthik Nandakumar. Diffusemix: Label-preserving data augmentation with diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 27621–27630, 2024.

[41] Xin Jin, Siyuan Li, Siyong Jian, Kai Yu, and Huan Wang. Mergemix: A unified augmentation paradigm for visual and multi-modal understanding. *arXiv preprint arXiv:2510.23479*, 2025.

[42] Siyuan Li, Zedong Wang, Zicheng Liu, Cheng Tan, Haitao Lin, Di Wu, Zhiyuan Chen, Jiangbin Zheng, and Stan Z. Li. Moganet: Multi-order gated aggregation network. In *International Conference on Learning Representations*, 2024.

[43] Lin Zhang, Zaixi Cheng, Ying Shen, and Dongqing Wang. Palmprint and palmvein recognition based on dcnn and a new large-scale contactless palmvein dataset. *Symmetry*, 10(4):78, 2018.

[44] Xingpeng Xu, Zhenhua Guo, Changjiang Song, and Yafeng Li. Multispectral palmprint recognition using a quaternion matrix. *Sensors*, 12(4):4633–4647, 2012.

[45] Pedro Tome and Sébastien Marcel. On the vulnerability of palm vein recognition to spoofing attacks. In *2015 International Conference on Biometrics (ICB)*, pages 319–325. IEEE, 2015.

[46] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[47] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

[48] Siyuan Li, Zedong Wang, Zicheng Liu, Di Wu, and Stan Z Li. Openmixup: Open mixup toolbox and benchmark for visual representation learning. *arXiv preprint arXiv:2209.04851*, 2022.

[49] Muzammal Naseer, Kanchana Ranasinghe, Salman Hameed Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Intriguing properties of vision transformers. In *Neural Information Processing Systems*, 2021.

[50] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.