Hindawi
International Journal of Aerospace Engineering
Volume 2022, Article ID 7868142, 12 pages
https://doi.org/10.1155/2022/7868142

Research Article

Continuous Gesture Sequences Recognition Based on Few-Shot Learning

Zhe Liu, Cao Pan, and Hongyuan Wang

Changzhou University, Changzhou 213000, China

Correspondence should be addressed to Zhe Liu; 18416120@smail.cczu.edu.cn and Cao Pan; pccczu@cczu.edu.cn

Received 28 June 2022; Revised 23 August 2022; Accepted 20 September 2022; Published 11 October 2022

Academic Editor: Qing Gao

A large number of demands for space on-orbit services to ensure the on-orbit system completes its specified tasks are foreseeable, and the efficiency and the security are the most significant factors when we carry out an on-orbit mission. And it can improve human-computer interaction efficiency in operations with proper gesture recognition solutions. In actual situations, the operations are complex and changeable, so the gestures used in interaction are also difficult to predict in advance due to the compounding of multiple consecutive gestures. To recognize such gestures based on computer vision (CV) requires complex models trained by a large amount of datasets, it is often unable to obtain enough gesture samples for training a complex model in real tasks, and the cost of labeling the collected gesture samples is quite expensive. Aiming at the problems mentioned above, we propose a few-shot continuous gesture recognition scheme based on RGB video. The scheme uses Mediapipe to detect the key points of each frame in the video stream, decomposes the basic components of gesture features based on certain human palm structure, and then extracts and combines the above basic gesture features by a lightweight autoencoder network. Our scheme can achieve 89.73% recognition accuracy on the 5-way 1-shot gesture recognition task which randomly selected 142 gesture instances of 5 categories from the RWTH German fingerspelling dataset.

1. Introduction

Space on-orbit systems are play a crucial role in aerospace. In relative independent working environments, problems and instrument malfunctions are not always foreseeable. In order to make the on-orbit system completes the target tasks, it is necessary to maintain the services and eliminate the malfunctions. Space on-orbit servicing (OOS) can be regarded as a suitable mode to maintain the on-orbit system and prolong its service life. In the process of space on-orbit service, frequent human-machine interaction operations are required, especially in the tasks such as orbit cleaning, service maintenance, or the cargo resupply [1]. The gesture is kind of complex rigid body which carries abundant human posture information, and the challenge of its state detection coexists with its practice ability in such interaction scene. The gesture recognition mainly has two parts applications in those on-orbit service tasks. One is to use specific gestures as a medium for signal transmission to manipulate machines such as multidegree of freedom robotic arm [2] by different gestures, which can improve the working efficiency and safety, and the other is to encode gestures for mapping different operations. Relevant research in gesture recognition field started earlier and achieved fruitful achievements. One direction to recognize gestures is by the helping of external sensor equipment [3] to gather joints' relative depth assisted with vision information, using methods such as scale invariant feature transform (SIFT) [4] to manually extract gesture features and combine them with feature classifier for recognition, and the other direction is to regression from the RGB images directly. However, to detect RGB images directly based on visual schemes often requires the construction of complex models and large amounts of datasets for training. With the development of

machine learning and computing hardware, the deep learning models with strong capacity of image feature representation such as convolutional neural networks (CNNs) [5] or transformer [6] achieved more than 95% recognition accuracy on 2D or 3D gesture recognition tasks. Compared with gesture image recognition, continuous gesture recognition requires the model with stronger ability of feature extraction to recognize from gesture sequences due to the blur by the motions. Thus, we usually extract instances' features by modeling the whole gesture sequence, such as using representative long short-term memory (LSTM) or dynamic time warping (DTW) methods in the task with the help of external sensors.

When the above models are dealing with the single gesture or continuous gesture recognition tasks, it often needs to input a plenty of gesture datasets that contain various backgrounds and gesture shapes to obtain better performance and generalization ability. Especially when the problem is extended to the recognition of continuous gestures, the scale of the hypothetical space increases rapidly, the cost of labeling datasets is hard to accept, and it is difficult to collect enough instances fit for task in some cases. But some studies have shown that [7] even if the sample in the task is insufficient, the model can also achieve excellent generalization performance and competent the classification or regression task by conduct proper few-shot learning method.

There are many ways to realize few-shot learning schemes. In the early stage, the classification and recognition task of few-shot learning is based on the models with relatively simple structure, such as nearest neighbor or linear support vector machine (SVM). Later, with the development of deep learning fields, transfer learning can also be used to synchronize the model parameters learned from large number of samples to other similar tasks and then fine tune the model parameters based on a small number of samples [8], which are an effective way for few-shot learning, but still need a considerable dataset for previous learning.

Aiming at the above problems of continuous gesture recognition, we propose a gesture recognition scheme based on small samples. Our scheme fine tunes the parameters of the self-attention (SA) module according to the support set provided by the task, using Mediapipe framework to detect the gesture landmarks from the input images, then split and extract those points' feature information of continuous gestures with the help of a lightweight autoencoder network, and output the class of the most similar instance in the support set as the result. Section 3 introduces the Mediapipe, the architecture of autoencoder network, and how we deconstruct the gesture key point information. Section 4 lists the experiments related to the parameter evaluation and the performance of the model.

2. Question Definition and Related Work

2.1. Question Definition. The continuous gesture recognition scheme conducted in this paper mainly aims at the few-shot learning tasks of recognize gesture sequences, and this kind of task only provides the support set and the query set. The support set contains few instances in each class for model training, and the query set contains many instances for prediction. The prediction of gesture essentially is a process of calculating the feature states of the input query set D_q and support set D_s, respectively, by the gesture feature extractor $f_{\mathrm{AE}}(D)$, then output the class $\hat{y}$ of the instance with the maximum cosine similarity in the support set as the prediction results of query instance (1).

$$\hat{y} = \max \left\{ \cos \left(f_{\mathrm{AE}}(D_{s,i}), f_{\mathrm{AE}}\left(D_q\right) \right) \right\} D_{s,i} \in D_s. \quad (1)$$

In practical problems, there will be considerable differences between gesture actions of different instances even they belong to the same class, which is mainly because the gesture actions in the switching process between key gestures in an instance are not constrained. It will cause large prediction deviation if we treat all those frames as the feature frames of continuous gestures. Therefore, we need to eliminate these frames that carry few gesture features to reduce the deviation in the prediction task.

2.2. Related Work

2.2.1. Hand Landmark Detect. Mediapipe [9, 10] is an open-source vision based real-time on-device hand tracking solution and has made many improvements for the real-time recognition on the premise of ensuring the recognition accuracy. Mediapipe divides gesture key point landmark detect task into two parts: palm detection and key points landmark detection. The palm detector reduces the image size by cut out the gesture part from the origin image to improve the efficiency of key point detection. This paper uses this framework to realize the task of gesture key point detection. Details of the gesture key point landmark detection are introduced in Section 3.4, and the mapping coordinates of gesture key points are shown in Figure 1.

With the help of recognition targets' inherent structure information, references [6] extract the gesture feature information directly without further data augmentation, which enlightened the gesture recognition scheme conducted in this paper. They proposed a nonautoregressive coding mechanism by making full use of the internal structure information of 3D gestures such as the interdependence between joints, which provides complete gesture information for the decoder while maintains the parallel structure. By taking the average distance error of joint points as the performance evaluation index, the nonautoregressive transformer model obtained 6.47, 7.55, and 9.80 distance errors on Imperial College Vision Lab (ICVL), Microsoft Research Asia (MSRA), and New York University (NYU) datasets achieved state-of-the-art performance on the 3D gesture recognition tasks.

2.2.2. Continuous Gesture Recognition. The continuous gesture recognition tasks with the external sensors' data, Zhang [11] expressed the gesture feature information with vectors consist of hand coordinates, interfinger distance, fingertip angle, and hand moving speed with the help of leap motion sensor [12] and achieved 98.50% recognition accuracy on the dataset containing 16 dynamic gestures.

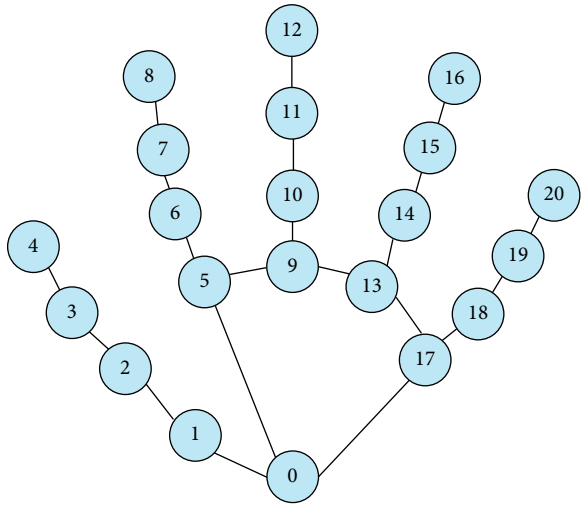

Figure 1: Gesture landmark mapping [9].

In reference [13], a small sample learning scheme for constructing convolutional neural network to analyze gesture sequence data returned by radar is proposed. Four basic gesture sequences were captured using frequency modulated continuous wave (FMCW) 60 GHz radar, and more than 94% recognition accuracy was achieved in the 5-way 1-shot task.

2.2.3. Image Superresolution. Due to the influence of recognition rate and motions, the resolution of the gesture part in the frame could reduce. For these changes in image resolution, we can recover information from those low-resolution images by some superresolution method before recognition. Deep convolutional neural networks (CNNs) have been adopted in superresolution widely. However, CNNs depend on its deep network structure to achieve better performance and often result any inconvenience in training such as instability or hard to converge. Coarse-to-fine super-resolution CNN (CFSRCNN) [14] are gathering complementary contextual information to overcome this issue and proposed a novel feature fusion scheme based on heterogeneous convolutions that achieved high efficiency of single image superresolution (SISR) without decrease the quality of reconstructed images.

Recent generative adversarial networks (GANs) also can help with those low-resolution image problems with small samples [15]. Enhanced superresolution group CNN (ESRGCNN) [16] is kind of flexible and efficient way in SISR. It balanced SISR performance and complexity by using group convolutional and residual learning techniques in group enhanced convolutional blocks and also used an adaptive up-sampling mechanism to make superresolution model more flexible in real tasks.

2.2.4. Transfer Learning. Transfer learning is mainly subsumed model transfer or data transfer, both of which have certain application value in small sample learning tasks. About model transfer scheme, a visual few-shot gesture recognition scheme is proposed in reference [8], which completes the recognition task by transferring the model parameters trained from a large number of datasets to another sampler but similar model. Specifically, train a GoogLeNet on Kungliga Tekniska Högskolan (KTH) dataset to extract gesture features, and use the probability network as the classifier to achieve 99.47% recognition accuracy on Keck gesture dataset. The few-shot continuous gesture recognition scheme in this paper uses similar strategies but uses a more lightweight feature extraction model pretrained on a small dataset. The few-shot recognition scheme in this paper also applies model transfer thoughts, pretrains the gesture feature extraction model from a self-made dataset, and then applies it to the new task support set and query set to extract the gesture feature information to realize the transfer of the model.

2.2.5. Fine Tuning. Reference [7] pointed out that using fine tuning strategy to build a small softmax network based on support set can improve the recognition accuracy of few-shot learning tasks by 2% to 7%. This paper also applies the fine tune strategy on the support set to improve the recognition accuracy of the model on specific task. The details of our fine tune method are described in Section 3.5.

3. Continuous Gesture Recognition

Our scheme mainly consists of two models to fulfill the work, the gesture key point landmark detector and the gesture feature extractor. To obtain the gesture key points' landmarks and relative depth information, we use Mediapipe gesture landmark detector to process the RGB-image sequences, and each frame outputs 63-dimensional vectors V_{lm}. Then decomposes the feature of vector V_{lm} into palm rotation feature and finger bending feature according to Section 3.1 and produces six 3-dimensional vectors V_{input}, the feature extraction model f_{AE} extracts the gesture feature data and outputs a 6-dimensional vector V_{output} as the gesture feature of the current frame. The model architecture is shown in Section 3.2. The vector is used to update the states of various gesture features in time sequence, and the mean value of each state feature is added to obtain a 6-dimensional vector V_{res} as the feature of this continuous gesture sequence instances. The detail of implementation process is described in Section 3.4, then output the instance's class with the highest cosine similarity in the support set as the result of prediction. The structure of the scheme is shown in Figure 2.

3.1. Gesture Information Decomposition. The gestures contain sufficient internal structure information such as joint attachment; thus, we decompose the gesture information before the recognition, so that the model can learn enough information even with few samples. The decomposition of gesture information is based on the 21 gesture key points' landmarks and relative depth information predicted by the gesture key point detection model. For any motions and postures of the palm, decompose them as follows:

(1) The curvature of five fingers is measured by the root of the same finger and the root of the palm at the distal, middle, and proximal phalanges of each finger

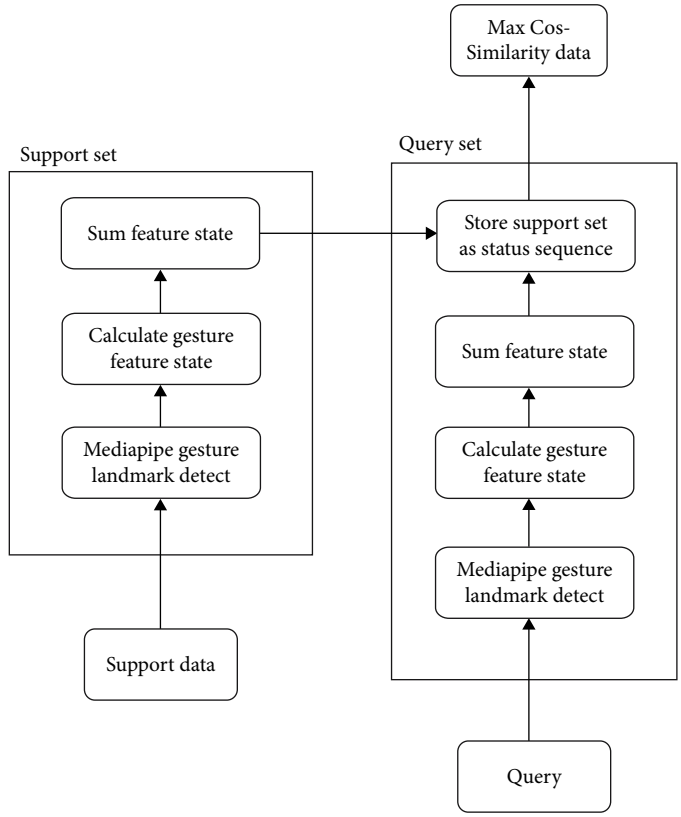

Figure 2: Structure of continuous gesture recognition scheme.

(2) The degree of palm turnover is measured by the distance between the keys at the base of the index and ring fingers

(3) The palm yaw degree is measured by the normalized vector from the palm root key to the middle finger root key

(4) The palm pitch degree is measured by the relative depth of the root of the middle finger to the palm root

The data in the gesture landmark vector V_{lm} are the fixed relative position; thus, there is no need to rescale the coordinates due to the different distance between the palm and the camera. The continuous gesture recognition method conducted in this paper divides the gesture feature into the above four basic parts.

3.1.1. Five Finger Joint Curvature Measurement. Due to the limitations of the palm joint, the fingers can only curl up and stretch in space. We use the five fingers bending degree information at any time, and the relative Euclidean distance between joint points is used to eliminate the influence of the palm rotation angle in space on the measurement of finger bending degree.

For any normal palm at any time, the included angle between the distal phalanx and the middle phalanx of the little thumb, ring finger, middle finger, and index finger is about 90-180 degrees; the included angle between the middle phalanx and the proximal phalanx is about 70-180 degrees; and the included angle between the proximal phalanx and the metacarpal bone is about 80-110 degrees. Although No. I metacarpal bone is more flexible than other four metacarpal bones, we assume that the angle between its proximal phalanx and metacarpal bone is roughly 90-180 degrees.

The gesture landmarks in this scheme are coordinates in three-dimensional space predicted by Mediapipe gesture landmark detector. Therefore, for the thumb, the Euclidean distance between the landmarks of the fingertip, middle segment, and No. I metacarpal bone and the landmarks at the root of the wrist is selected as the measurement of the bending degree (4). The other four fingers' bending degree consist of three parts' European distance, the top of distal phalanx and the top of proximal phalanx, the top of middle phalanx and the bottom of proximal phalanx, and the top of proximal phalanx and the wrist root. Using the embedded vector V_{lm} of five finger landmark data in space output by Mediapipe, calculate the five fingers' feature vector of the bending degree by (2).

3.1.2. Palm Turnover Measurement. In this scheme, the Euclidean distance between the II and IV metacarpal bones after rescaling is selected as the measurement of the degree of palm rotation. The landmarks of the gestures' key points corresponding to the II and IV metacarpal bones are 5 and 13, and the subscripts in the feature dataset are 13 and 37.

The Euclidean distance between them is calculated to get the result of the palm turnover's degree (2).

$$V_{\text{input}}[15] = \sqrt{\left(V_{lm}^{(15)} - V_{lm}^{(39)}\right)^2 + \left(V_{lm}^{(16)} - V_{lm}^{(40)}\right)^2}. \quad (2)$$

The formula that measures palm turnover's degree is continuous, monotonic, and differentiable in a rotation cycle; the degree range is between -0.06 and 0.06. The measurement value obtained by the counterclockwise rotation of the palm at the top angle increases from -0.06 to 0.06, and the measurement value obtained by the clockwise rotation decreases from 0.06 to -0.06, stripping the influence of the yaw and pitch angle on the measurement turnover.

3.1.3. Palm Pitch Measurement. The pitch degree of the palm in space is measured by the relative distance between the root of the middle finger and the root of the palm. The subscripts of those two points in the landmark map are 9 and 0. The measurement result of the pitch degree is shown in the following equation.

$$V_{\text{input}}[16] = \left(V_{lm}^{(38)} - V_{lm}^{(3)}\right). \quad (3)$$

3.1.4. Palm Yaw Measurement. The palm yaw degree is measured by the two-dimensional normalized vector of the palm root point to the middle finger root in space. This measurement method can eliminate the influence of the other two spatial angles on the measurement of palm yaw angle. Palm yaw angle is measured by the following equations.

$$\rho = \sqrt{V_{\text{input}}[17]^2 + V_{\text{input}}[18]^2}, \quad (4)$$

$$V_{\text{input}}[17 + i] = \frac{\left(V_{lm}^{(27+i)} - V_{lm}^{(i)}\right)}{\rho}\ i \in \{0, 1\}. \quad (5)$$

After the decomposition of gesture feature information, we get six vectors to preliminary describe the gesture in each frame. Then embedded them into a 17-dimensional vector V_{input} according to the subsequence of bending degree of thumb, index finger, middle finger, ring finger, little finger, and the palm pose vectors, the embedding map is shown in Figure 3. The embedded vector V_{input} cannot represent the gesture feature directly, so we need to further extract features from it.

3.2. Architecture. The model used to extract the decomposed gesture features in the few-shot learning task is a pretraining model combined multiple autoencoders with a self-attention module [17]. The goal of pretraining is to use an adequate training set including different actions and postures palm to get a lightweight model with strong generalization ability, so that we can fine tune the parameters by different tasks to improve model's performance a step further. Therefore, we added a self-attention mechanism in front of the encoder to extract content information in specific tasks [18]. Initially, we only train the autoencoder module to extract single gesture feature form the training set that independent of query set, when it is applied to the real tasks, then train the self-attention module's parameters by the support set to focus on the context information of continuous gestures, which is an effective way to improve the recognition accuracy of the model for the current task.

During prediction, the model is used to calculate the features of various samples in the support set, extract the feature information of instances in the query set, and then extract the feature state and eliminate the impact of gesture action change frames on the overall recognition task. Finally, calculate the cos similarity and output the class of the highest similarity instance in support set as the prediction result.

3.2.1. Autoencoder. The distribution of various gesture feature data obtained from the above decomposition in the original space is linear inseparable. It will loss a large amount of information if we use linear feature extraction method. Therefore, our model relies on the autoencoder to compress the vector dimension to extract gesture features. Section 4 shows relevant experiments. The input of the model is the decomposed 17-dimensional vector V_{input}. Since the coordinate value range of gesture key points recognized by Mediapipe is stored as a percentage value of the input image boundary according to the distance, it is easy to cause gradient disappearance if we use V_{input} for model training without any process. We use (6) to process the datasets with layer normalization firstly, where γ is the offset value and E is the mean value of V_{input}.

$$V_{\text{input}}[i] = \frac{V_{\text{input}}[i] - E\left(V_{\text{input}}\right)}{\sqrt{\text{Var}\left(V_{\text{input}}\right) + \gamma}}\ i \in N. \quad (6)$$

The standardized data is divided into six 3-dimensional vectors V_{di} according to the embedding method mentioned above, which is, respectively, transmitted to different autoencoders. Each encoder is composed of two hidden layers and independent of each other, and they have different parameters and connected to a decoder, respectively. The decoder is used to restore the output value of the encoder back to the input vector value. Calculate the loss function according to the decoders' results, and iteratively train the parameters of each batch. The strategy of training is introduced in Section 3.6. The bending degree feature of the thumb is extracted from the encoder subscript from 0 to 4, and the palm posture feature is extracted from the encoder subscript 5. When extracting features, the model only calculates the values of each encoder and embeds them into 6-dimensional vector as output. The feature extractor model structure is shown in Figure 4.

The 6-dimensional vector V_{output} indicates the gesture feature of the current frame. There are six elements in the embedded vector. The top five are the feature values of each finger's bending degree, and the last is the posture feature of the palm. The extracted gesture feature vectors of the whole frames are input into the state extraction module to eliminate the impact of gesture frames that did not carry adequate feature information on the gesture recognition task.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Thumb bending degree			Index finger bending degree			Middle finger bending degree			Ring finger bending degree			Pinky finger bending degree			Hand pose			

Figure 3: Feature vector embedding mapping.

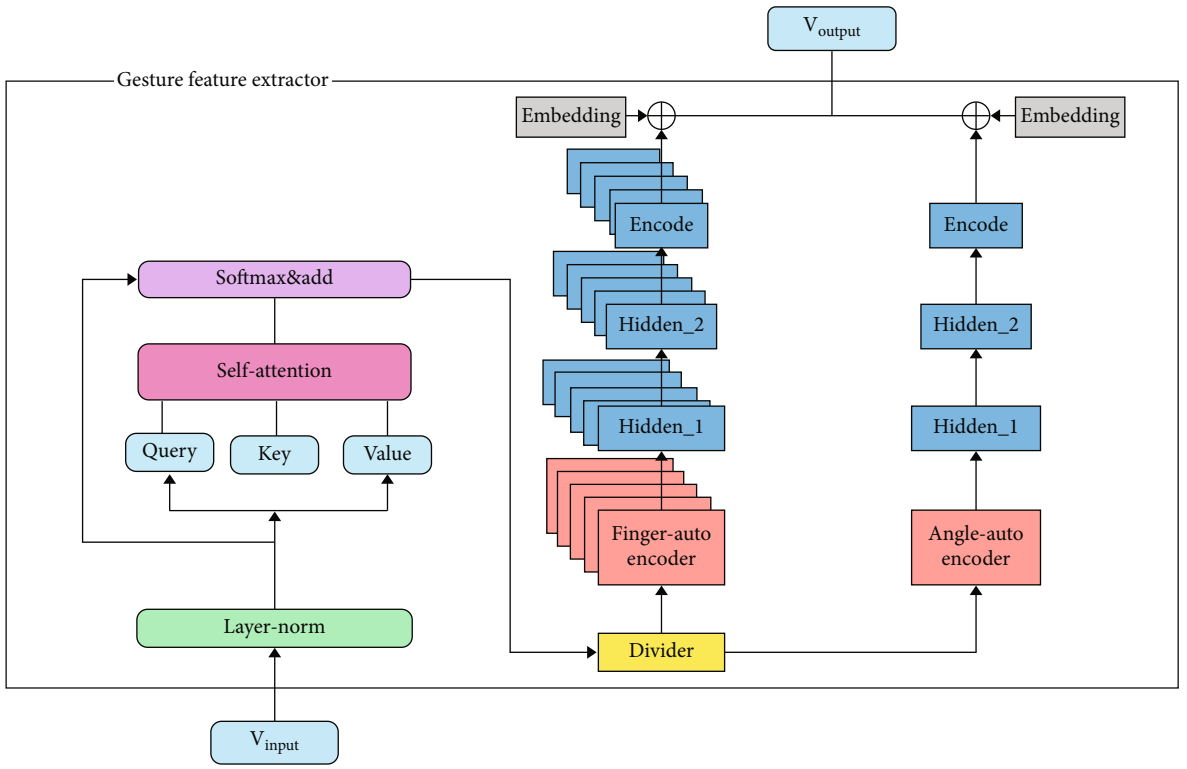

Figure 4: Architecture of the gesture feature extractor.

3.3. State Extraction of Sequences. For different video instances that belong to the same gesture sequence, the gesture states between instances may be misaligned due to the different duration time of a gesture state or the speed of the gesture transformation, and Figure 5 shows the results of using the pretraining model to extract two instances in the support set and the test set that belong to the same class. The test set instance lags behind the support set instance by about 20 frames. And there will be a large gap if we calculate the similarity between the two instances' feature sequences directly. Therefore, we propose a scheme of detecting and matching instances by extract the gesture feature state.

The feature states in this paper refer to the values of the vector V_{output} sequence that are maintained within a given threshold and stable for a certain frame length $\text{Frame}_{\text{len}}$. The method of state extraction receives the V_{output} vector and maintains the mean value avg of six current states. Whenever a new V_{output} is input, check whether the difference between the data in the vector and the mean value avg of their respective states is greater than the given threshold, and it is considered to be the end of a state when the avg is greater than the threshold value. Then check whether the

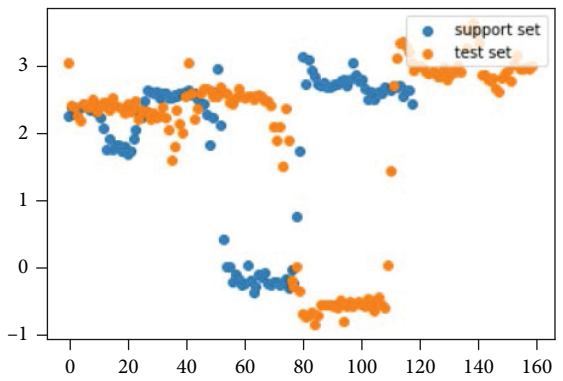

Figure 5: Example of gesture feature dislocation.

frame length $\text{Frame}_{\text{len}}$ reaches the minimum state frame length. Only when it is reached, record the mean value avg of the current state and reset the frame length. Otherwise, update the status mean value by the following equation.

$$\text{avg}_i = \frac{\text{avg}_i * \text{Frame}_{\text{len}} + V_{\text{input}}}{\text{Frame}_{\text{len}} + 1} \; i \in \{0, 1, 2, 3, 4, 5\}. \quad (7)$$

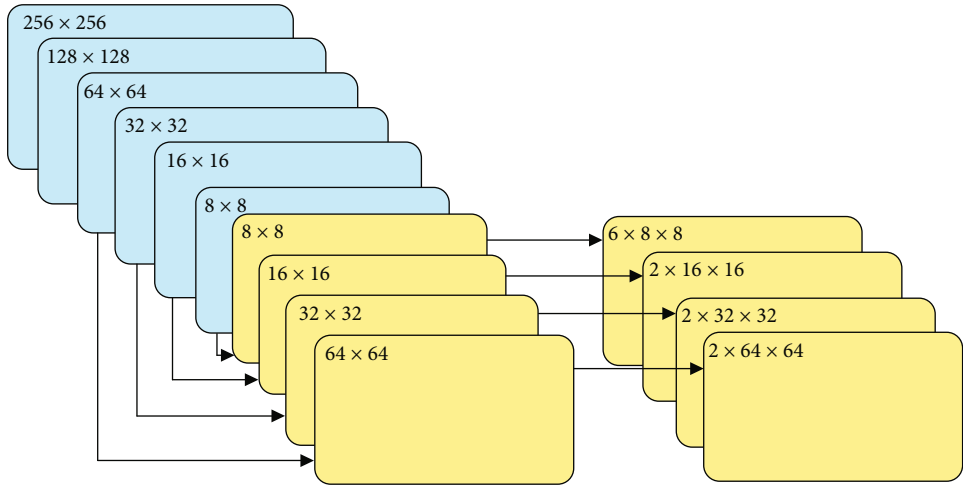

Figure 6: Structure of palm detector [9]. The blue squares are the encoders, and the yellow squares are the decoders.

Using above method to process feature vector of each instance, add the state feature of each dimension record of the instance to obtain a 6-dimensional vector. The matching between different instances is to calculate the similarity between vectors. In this paper, the cosine similarity between the two vectors is selected as the degree of similarity. For instances in the query set, the class of the instance with the greatest cosine similarity is in the support set as the prediction result.

3.4. Gesture Key Landmark Detect. In order to solve the large-scale problem of continuous gesture recognition, we first need to detect the key landmark of gestures in the image. In this paper, Mediapipe is used to detect the 21-key landmark of gestures in RGB images. The model inputs real-time RGB image data and returns the relative coordinates and depth of 21 hand key points in the image. The Mediapipe can achieve more than 30 frame rates in real-time processing tasks when the computing power is sufficient. The core framework of Mediapipe is implemented by C++. The model is mainly divided into two parts. First, the palm detector, which replaces the hand detector, uses a hand positioning frame with direction information to locate the position of the palm in the image. The second is the hand coordinate model, which obtains the coordinate data by detecting the gesture key landmark in the positioning frame processed by the palm detector.

3.4.1. Palm Detector. It is built by a relatively lightweight convolutional neural network with encoding and decoding structure. The palm part or the fist is a rigid object that is easy to recognize, thus to detect the palm or fist part from the input images instead of the whole hand. The palm detector receives the complete image as the input, cut out the palm part from the image when the palm is detected, and then input the result into landmark detector for regression, which greatly reduces the irrelevant contents from the origin image. The cut palm boundary box is square; compared with other shapes, it can reduce the proportion of useless

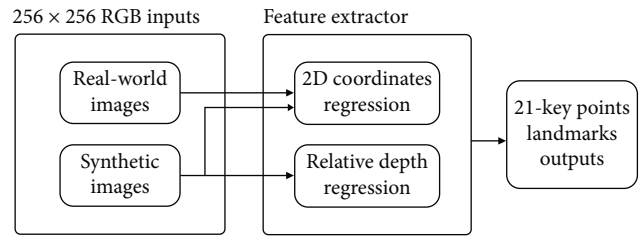

Figure 7: Structure of gesture landmark detector [9].

Table 1: Model performance on RWTH dataset.

Model	Similarity measure	Fine tuning	5-way 1-shot acc	5-way 3-shot acc
Full	MSE	N	77.40%	78.39%
Full	Cosine	N	86.67%	88.20%
Full	MSE	Y	79.45%	80.72%
Full	Cosine	Y	89.73%	90.16%

Table 2: Comparison with other models.

Model	5-way 1-shot accuracy
Nearest neighbor	75.81%
Nearest neighbor (SE)	77.45%
Matching network [20]	85.62%
$\text{CNN}(64)_{\times 4}$ [18]	80.95%
Autoencoder (ours)	89.73%

Table 3: Model performance on Sebastien Marcel dynamic dataset.

Model	Similarity measure	Fine tuning	4-way 1-shot acc	4-way 3-shot acc
Full	MSE	N	89.62%	90.41%
	Cosine	N	91.77%	91.73%
Full	MSE	Y	91.14%	91.11%
	Cosine	Y	92.17%	93.23%

Table 4: Influence of linear and nonlinear dimensionality reduction methods on model recognition accuracy.

Dataset	Similarity measure	PCA 4-way 1-shot	PCA 4-way 3-shot	Autoencoder 4-way 1-shot	Autoencoder 4-way 3-shot
RWTH	MSE	73.48%	74.82%	79.45%	80.72%
	Cosine	68.67%	70.16%	89.73%	90.16%
SMD	MSE	74.62%	74.47%	91.14%	91.11%
	Cosine	71.53%	74.13%	92.17%	93.23%

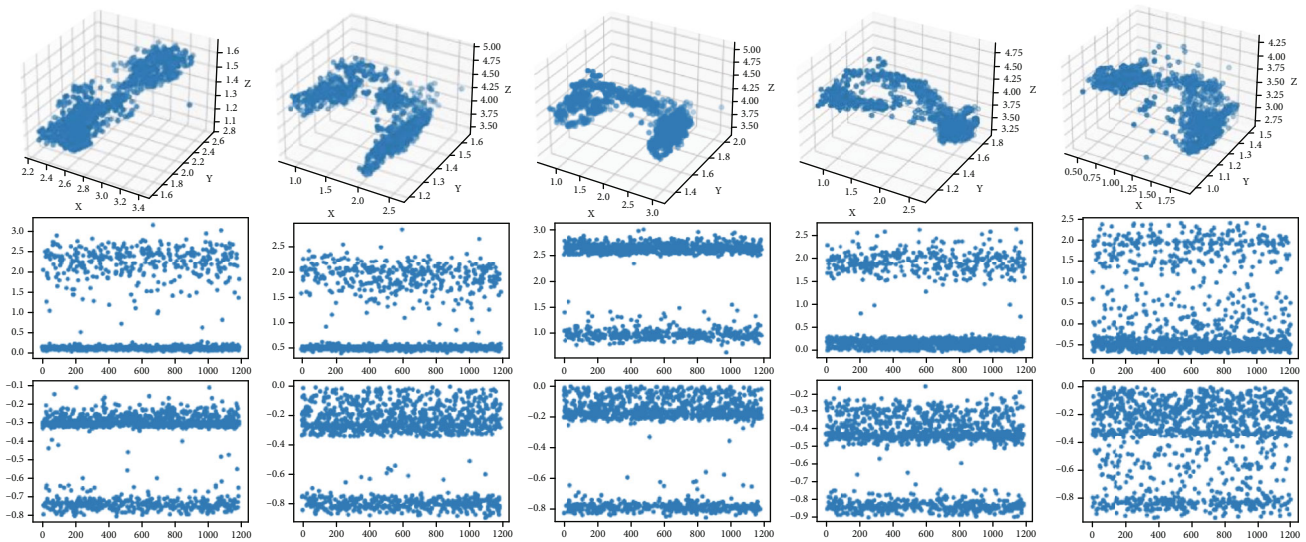

Figure 8: Effects of different dimensionality reduction methods on feature extraction.

information by 3 to 5 times and improve the efficiency of gesture key coordinate detection.

The encode-decode structure in the palm detector can extract the nonlinear features of sample instances through iterative training. The important thing is that the feature receptive field in large-scale images is relatively small, while the feature receptive field in small-scale images is relatively large. Such a structure can enable the model to perceive the feature information in images of different sizes. The focal loss [19] function is used in the model training. The ablation experiment designed in the paper found that the cross entropy loss function with encoder has a better accuracy on the dataset than that without encoder. The average accuracy of the cross entropy loss function without encoder is 86.22%, and that with encoder is 94.07%. The palm detector model structure is shown in Figure 6.

3.4.2. Gesture Landmark Detector. The input data in this part is a smaller image cut by the palm detector, which allows the model to focus on the 21-key point coordinate regression task. The dataset used in the model training includes not only the palm image data in different backgrounds from the real world but also the synthetic gesture data. On the one hand, the synthetic data provides the accurate position information of gesture key points in space, and landmark detector can learn additional relative depth information under the supervision of such data, hence, to improve the performance on the key points' coordinate regression task. On the other hand, in the actual gesture image, due to the high degree of freedom of the gesture itself and various occlusion problems, the accurate position information carried by such data can well enable the model to learn the gesture information of the occluded part. Reference [9] shows that model trained by datasets combined real-world and synthetic gestures performed better than only trained by real-world or synthetic gestures and achieved 13.4% mean-square error (MSE). The main structure of gesture key point detector is shown in Figure 7.

The 2.5D fidelity data obtained by the landmark detector include that the horizontal and vertical coordinates of the key landmark in the RGB images and the relative depth data between each key landmark at the root of the palm. Each time the model receives an image frame as input, cut it into a small-scale image that containing the complete hand by the palm detector and then input it into the key point coordinates detector to detect the three values of 21 key landmark points. Each frame of gesture image outputs 63 data as V_{lm} vector.

3.5. Fine Tuning. The fine-tuning strategy in few-shot learning often means to fine tune the model parameters according to the prediction results of few sample instances in the actual

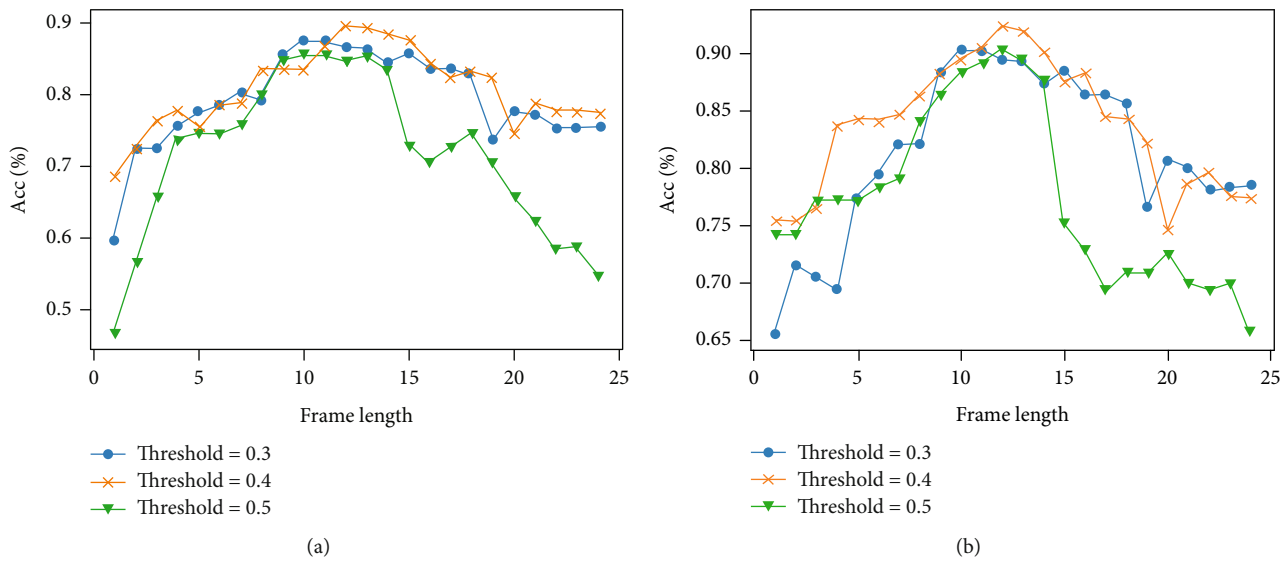

Figure 9: Effects of different threshold and $\text{Frame}_{\text{len}}$ on model recognition accuracy.

task after the pretraining. This strategy is often used to optimize the model of the transfer learning network. Experiments show that using this strategy to optimize the few-shot learning model can improve the prediction accuracy by 2-7% [7].

Because the number of instances in the support set is small, it is not suitable for training new classifiers, also the overall structure of the pretraining model is very light. Therefore, our fine-tuning strategy of the model is only to fine tune the parameters of the self-attention module based on the support set instances and introduce different attention scores for different feature vectors. In order to prevent the model from over fitting, a smaller iteration number and a larger iteration step are selected to fine tune the model. Experiments show that this fine-tuning strategy can improve the recognition accuracy of the model by 2-4%.

3.6. Training. The model is mainly training two parts: the autoencoder and the self-attention module. The dataset used for training autoencoders that extract bending degree contains the gesture key landmark information extracted from 11 RGB videos collected by 2 people who make palm grasping and opening movements under different backgrounds, a total of 1186 instances of training set, 20 pieces of which are randomly selected and divided into a batch. The dataset of the training palm pose feature extraction autoencoder contains the gesture key points extracted from 12 RGB videos collected by 2 people that make various gesture under different backgrounds.

The pretraining model uses empirical risk minimization strategy for training, in which six autoencoders are independent of each other, each learns feature information from the V_{input} set and select the cosine similarity between the decoder results and the instances as the loss function. We use Adam optimization algorithm and initial learning rate to 0.01. Each encoder iterates 20 batches of training sets.

This paper uses the one-head attention module of dot product attention mechanism. In specific tasks, the model uses the support set to train the W_q, W_k, and W_v linear transformation layers in the self-attention module as the fine-tuning strategy. Out of the residual structure in the self-attention module, the weights of the above linear are initialed to zero, which avoid to affect the training of the autoencoders. The stochastic gradient descent (SGD) optimizer is used to train the module parameters, the initial learning rate is set to 0.01, and the momentum is set to 0.9. The cosine similarity between the current frame and other frame features is used as the loss function for training. Each training set contains various randomly selected support set instances, and a total of 5 batches of data are iterated in the training process.

4. Experiment

This part shows some problems encountered in the process of model conduction, the selection of the recognition scheme, and the optimization of details and also includes the experiments to support the solutions we choice. Sections A and B mainly include the experiments about the performance of our method on different datasets, the selection of feature extraction methods for the deconstructed gesture information, different few-shot models' performance on the gesture recognition task, and the comparison of several linear and nonlinear feature extraction schemes. Section C introduces the experiment of feature state extraction of continuous gestures and specifically compared the recognition accuracy of different thresholds and continuous steps on the recognition task. Section D introduces the contribution of different autoencoders and the self-attention module to the accuracy of model recognition. We validated the result by an ablation experiment. The experiments above are mainly conducted on the RWTH German fingerspelling and the Sebastien Marcel dynamic public datasets.

Table 5: Ablation study of the gesture feature extractor model.

Model	Similarity measure	RWTH 5-way 1-shot acc	RWTH 5-way 3-shot acc	SMD 4-way 1-shot acc	SMD 4-way 3-shot acc
Full	MSE	79.45%	80.72%	91.14%	91.11%
	Cosine	89.73%	90.16%	92.17%	93.23%
None AE_5	MSE	79.41%	80.47%	90.36%	91.34%
	Cosine	85.62%	85.95%	91.69%	92.10%
None AE_4	MSE	75.71%	75.64%	89.97%	89.84%
	Cosine	82.88%	84.13%	90.38%	90.75%
None AE_3	MSE	75.34%	77.35%	86.79%	88.19%
	Cosine	84.25%	85.13%	90.20%	91.73%
None AE_2	MSE	79.42%	78.68%	77.69%	77.42%
	Cosine	88.36%	89.25%	81.99%	83.22%
None AE_1	MSE	72.60%	74.26%	76.40%	77.46%
	Cosine	85.62%	85.45%	80.40%	82.31%
None AE_0	MSE	74.66%	74.24%	81.18%	82.54%
	Cosine	86.30%	87.83%	86.27%	87.22%
None SA module	MSE	76.36%	77.69%	88.18%	88.78%
	Cosine	88.54%	88.83%	90.97%	91.02%

4.1. Results

(1) RWTH German fingerspelling dataset

The RWTH German fingerspelling dataset contains 35 different continuous gestures. Each gesture contains 88 video instances of stand 2 and stand 1, hereinafter, referred to as the RWTH dataset. Five continuous gestures are randomly selected in this experiment. Each gesture contains 22 video instances of stand 1 as the support set and query set. The model state threshold and continuous frame length are set to 0.4 and 14, and the average prediction accuracy of the model is obtained by repeatedly selecting 5 prediction categories. The prediction results are shown in Table 1. The table shows that the strategy of using the fine tune attention module in 1-shot or 3-shot recognition tasks can improve the recognition accuracy by 1.96% to 3.06% and achieve better performance by using cosine similarity measure the distance between feature vectors instead of mean square error (MSE).

Table 2 shows the results of various gesture feature extraction methods performing 5-way 1-shot tasks on the RWTH dataset. The nearest neighbor model selects a support instance nearest to the instance as the classification result and uses five gestures randomly selected in the dataset, one instance of each gesture as the support set for training. At the same time, the SE method is used to process the data to improve the model recognition accuracy by 1.64%. Matching network [20] is a model combining shallow CNN and bidirectional LSTM network. $\mathrm{CNN}(64)_{\times 4}$ in the table is a shallow network composed of 4 convolution layers with 64 channels, which is trained by LSTM meta-learner method [21, 22]. According to the prediction results of this task, the method proposed in this paper obtains extra information by dividing gesture feature, thus achieves higher recognition accuracy than other small sample learning methods.

(2) Sebastien Marcel dynamic (SMD) dataset

The dataset contains four consecutive gestures: clip, rotate, stop grasp OK and No, each gesture records the action of 10 people as samples. Table 3 shows the influence of different factors on the recognition accuracy such as similarity measurement, fine-tuning strategy, and the number of instances in each class. Without using fine-tuning strategy to training self-attention module, the accuracy of the model achieves 1% to 3% higher when we use cosine similarity as the similarity measurement than using MSE. The recognition accuracy increases after fine-tuned self-attention module's parameters, and the 1-shot prediction task using cosine similarity measurement method increased 2.54%, which improved most obviously. The recognition accuracy of model using cosine similarity was improved 2.33% by fine-tuning strategy in the 3-shot task, which was more significant than that using MSE as similarity measurement.

4.2. Gesture Feature Extraction. The extraction of gesture features is to reduce the dimension of the basic gesture feature vector V_{input} decomposed by the method above. The experiments in this section show the impact of linear and nonlinear dimensionality reduction methods on the performance of the model and calculate the recognition accuracy of the feature extractor under the principal component analysis (PCA) and autoencoder extraction methods on RWTH and Sebastien Marcel datasets. Table 4 shows that the recognition accuracy

obtained by using the autoencoder method is 10-20% higher than that obtained by PCA on the two datasets.

Figures 8(a)–8(e), respectively, show the feature extraction results of different dimensionality reduction methods on the original data. The first row in the figure shows the distribution of the original data in the three-dimensional space, and the second and the third rows are the distribution maps reduced to 1-dimensional using autoencoder and PCA. Although it is shown in the figure that the within class distance of each eigenvalue obtained by the two dimensionality reduction methods is also small, the distribution of the original data is nonlinear separable, so the line dimensionality reduction method loses more information in the process of dimensionality reduction, which get the inferior accuracy of the prediction.

4.3. State Extraction of Sequences. This section mainly shows the impact of different thresholds and $\text{Frame}_{\text{len}}$ parameters used in the state extraction method on the accuracy of model recognition. The threshold values of 0.3, 0.4, and 0.5 were selected, respectively, to test the model recognition accuracy of the minimum state frame length $\text{Frame}_{\text{len}}$ from 1 to 25 on the RWTH dataset and the SMD dataset. Figure 9(a) shows the experimental results on the RWTH dataset. When the threshold is 0.4 and the $\text{Frame}_{\text{len}}$ is 16, the model achieves the highest recognition accuracy of 91.33%. Figure 9(b) shows the experimental results on the SMD dataset. When the threshold is 0.4 and the frame length is 11, the model achieves the highest recognition accuracy of 92.13%.

4.4. Continuous Gesture Recognition. In order to show the influence of each component in the model on the overall recognition results, we constructed an ablation experiment to show the contribution of each autoencoders and the self-attention module to the model recognition accuracy. The experimental results are shown in Table 5.

Table 5 shows the recognition accuracy of the model on RWTH and SMD datasets with the removal of each autoencoder module or the self-attention module. The influence of each module on the test set's recognition accuracy sampled from RWTH dataset ranges from 1% to 7%, among which the ring finger bending encoder AE_3 contributing 6.85% accuracy of recognition. On the SMD dataset, the influence of each module on the overall recognition accuracy of the model ranges from 1% to 12%. Among them, AE_1 improved 11.77% accuracy in the recognition task, which has the greatest contribution.

5. Summary

This paper conducts a few-shot gesture recognition scheme combined a gesture landmark detector with a lightweight gesture feature extractor. The scheme uses Mediapipe gesture key landmark detection model to recognize gesture key landmark points on RGB image sequences. The recognition results are processed into a 6-dimensional feature vector by the gesture feature extractor. Various feature states in the sequence are extracted and added and then output instances' class with the highest cosine similarity in the support set as the result. In the 5-way 1-shot tasks on RWTH and Sebastien Marcel datasets, the recognition accuracy was achieved 89.73% and 92.17%, respectively.

Also, the gesture feature state extraction scheme in this paper is mainly applicable to the recognition task of a single palm in gesture sequences. We can use the multihead attention structure to extract different association features according to the different context associations in the gesture sequences, to improve the recognition performance of the model in those continuous gesture recognition tasks with strong context correlation, such as sign language recognition.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interests.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (61976028).

References

[1] J. P. Davis, J. P. Mayberry, and J. P. Penn, "On-orbit servicing: inspection repair refuel upgrade and assembly of satellites in space," *The Aerospace Corporation, Report*, vol. 60, 2019.

[2] J. Herron, D. Lopez, J. Jordan et al., "RGB-D robotic pose estimation for a servicing robotic arm," 2022, http://arxiv.org/abs/2207.11537.

[3] E. Rahimian, S. Zabihi, A. Asif, D. Farina, S. F. Atashzar, and A. Mohammadi, "FS-HGR: few-shot learning for hand gesture recognition via electromyography," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 29, pp. 1004–1015, 2021.

[4] L. Jing, "Image feature matching based on improved SIFT algorithm," *International Journal of Circuits, Systems and Signal Processing*, vol. 12, 2018.

[5] A. Mujahid, M. J. Awan, A. Yasin et al., "Real-time hand gesture recognition based on deep learning YOLOv3 model," *Applied Sciences*, vol. 11, no. 9, p. 4164, 2021.

[6] L. Huang, J. Tan, J. Liu, and J. Yuan, *Hand-Transformer: Non-Autoregressive Structured Modeling for 3D Hand Pose Estimation*, Springer, Cham, 2020.

[7] G. S. Dhillon, P. Chaudhari, A. Ravichandran, and S. Soatto, "A baseline for few-shot image classification," 2019, http://arxiv.org/abs/1909.02729.

[8] R. Cheng, *Research and Application of Small Sample Gesture Recognition Based on Convolutional Neural Network*, Taiyuan University of technology, 2020.

[9] F. Zhang, V. Bazarevsky, A. Vakunov et al., "Mediapipe hands: on-device real-time hand tracking," 2020, http://arxiv.org/abs/2006.10214.

[10] C. Lugaresi, J. Tang, H. Nash et al., "Mediapipe: a framework for building perception pipelines," 2019, http://arxiv.org/abs/1906.08172.

[11] P. Zhang, *Research on Recognition Based on Structural Features and Deep Learning*, Henan University of science and technology, 2019.

[12] J. Cheng, *Dynamic Gesture Recognition Based on the Leap Motion Controller*, Jiangsu University of Science and Technology, 2021.

[13] G. Mauro, M. Chmurski, M. Arsalan, M. Zubert, and V. Issakov, *One-Shot Meta-Learning for Radar-Based Gesture Sequences Recognition*, Springer, Cham, 2021.

[14] C. Tian, Y. Xu, W. Zuo, B. Zhang, L. Fei, and C. W. Lin, "Coarse-to-fine CNN for image super-resolution," *IEEE Transactions on Multimedia*, vol. 23, pp. 1489–1502, 2021.

[15] C. Tian, X. Zhang, J. C. W. Lin, W. Zuo, and Y. Zhang, "Generative adversarial networks for image super-resolution: a survey," 2022, http://arxiv.org/abs/2204.13620.

[16] C. Tian, Y. Yuan, S. Zhang, C. W. Lin, W. Zuo, and D. Zhang, "Image super-resolution with an enhanced group convolutional neural network," 2022, http://arxiv.org/abs/2205.14548.

[17] A. Vaswani, N. Shazeer, N. Parmar et al., "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[18] Z. Ji and X. Chai, "Few-shot learning based on self-attention and auto-encoder," *Journal of Tianjin University: Natural science and Engineering Technology Edition*, vol. 67, 2021.

[19] T.-Y. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," 2017, http://arxiv.org/abs/1708.02002.

[20] O. Vinyals, C. Blundell, T. Lillicrap, and D. Wierstra, "Matching networks for one shot learning," 2017, http://arxiv.org/abs/1606.04080.

[21] Q. Cai, Y. Pan, T. Yao, C. Yan, and T. Mei, "Memory matching networks for one-shot image recognition," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, 2018.

[22] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning," in *International Conference on Learning Representations*, Palais des Congrès Neptune, Toulon, France, 2017.