Continuous Sign Language Recognition Via Temporal Super-Resolution Network

Qidan Zhu¹ · Jing Li¹ · Fei Yuan² · Quan Gan¹

Received: 7 September 2022 / Accepted: 15 February 2023 / Published online: 6 March 2023
© King Fahd University of Petroleum & Minerals 2023

Abstract
Aiming the problem that the spatial-temporal hierarchical continuous sign language recognition (CSLR) model with video as input is computationally intensive, thus limiting the real-time application, this paper proposes a temporal super-resolution network (TSRNet) to reduce the model computation while keeping the loss of accuracy to a minimum, achieving the best compromise between the real-time performance and accuracy. The TSRNet-based CSLR constructed in this paper consists of three main parts: frame-level feature extraction, temporal feature extraction and the proposed TSRNet, where the TSRNet is located between them, and consists of two branches: detail and coarse descriptors. The extracted frame-level features are first sparse, after which they are passed through the two branches designed for feature reconstruction; the fused dense sequence is subjected to temporal feature extraction. In order to better recover the semantic-level information, this paper also proposes a self-generating adversarial network training method, which treats the TSRNet as the generator and the frame-level and temporal processing parts as the discriminator. In addition, to unify the criteria for judging the loss of model accuracy under different benchmarks, this paper proposes word error rate deviation (WERD), where the error rate between estimated WER and reference WER obtained by reconstructed frame-level feature sequence and complete original frame-level feature sequence, respectively. Experiments on two large-scale sign language datasets demonstrate the effectiveness of the model. The method proposed in this paper is not only applicable to CSLR, but is general to spatial-temporal hierarchical models where the input is video data. Code is available at https://github.com/woshisad159/CSLR.git.

Keywords Continuous sign language recognition · Model real-time performance · Temporal super-resolution network · Self-generating adversarial training method · Word error rate deviation

1 Introduction
Sign language is a basic communication tool between normal people and hearing impaired people or within hearing impaired people. It has its own vocabulary, just like normal language, and in the process of communication using sign language, a gesture or set of gestures (hand patterns) is used as a bridge of information to convey lexical information between people, which plays an important role in practical life [1–4].

Video-based CSLR research is to convert sign language videos into continuous sign language vocabulary [5,6]. Due to the huge cost of frame-level annotation when creating continuous sign language video datasets, continuous sign language datasets often only have video-level annotations but no frame-level annotations, which is why researchers usually regard CSLR as weakly supervised learning [7]. In view of the fact that the dataset has no frame-level annotations in CSLR, many deep learning models have been proposed and applied in CSLR [8–11]. According to the methods of spatial-temporal feature extraction in these models, we divide the deep learning-based models for CSLR into two classes: One is the spatial-temporal hierarchical model, which first

¹ College of Intelligent Systems Science and Engineering, Harbin Engineering University, Harbin 150001, China
² Northwest Institute of Mechanical and Electrical Engineering, Xianyang 712099, China
extracts feature information from frame-level images, then extracts temporal feature information based on continuous frame-level feature sequences and finally performs classification. The other is the non-spatial-temporal hierarchical model, which extracts spatial and temporal feature information directly from the video for classification.

In the spatial-temporal hierarchical model, the extraction of spatial and temporal information from continuous sign language data is performed in series. Usually, convolutional neural network (CNN) [12] is used for spatial embedding, high-dimensional feature information is extracted from low-dimensional image information, and then recurrent neural network (RNN) [13], transformer [14] or long short-term memory (LSTM) [15] is used for processing in the temporal dimension to obtain high-dimensional sparse semantic information for final recognition and classification. The spatial-temporal hierarchical model has the characteristics of simple model, few parameters and clear layers, etc. It is the mainstream direction of model research in CSLR at present. However, because the spatial and temporal information is extracted and fused separately, there will inevitably be a loss of spatial and temporal information in this process, which will affect the final recognition accuracy [16].

In the non-spatial-temporal hierarchical model, the spatial and temporal information extraction of data is carried out in parallel, that is, the data are processed in the spatial and temporal dimensions at the same time, usually using 3D-CNN [17,18], 2+1D-CNN [19], spatial-temporal transformer [20] and other methods. The non-spatial-temporal hierarchical model can obtain more spatial-temporal information, but its model parameters are large and the model is cumbersome [8]. Since reducing the model calculation amount when the model remains unchanged will inevitably lead to the loss of the accuracy of the model, this paper aims to reduce the model calculation amount within a certain accuracy loss range and improve the real-time performance. This paper is a research improvement on the spatial-temporal hierarchical CSLR model, and in our research on the spatial-temporal hierarchical model, we found that:

1. The images between adjacent frames have high similarity in content. As shown in Fig. 1, the autocorrelation matrix is calculated and visualized after frame-level feature extraction for frame-level images, and the generated autocorrelation matrix has high similarity between adjacent feature vectors.

2. In the spatial-temporal hierarchical model, most of the computation of the model is concentrated on the extraction of video frame-level features, and the feature information extraction for each frame of the video is independent of each other. Therefore, this paper reduces the computational complexity of the model by reconstructing the sparse data into dense feature sequences while keeping the original spatial-temporal hierarchical model unchanged.

In this paper, a TSRNet is proposed to reconstruct the sparse feature sequence into a dense feature sequence, which mainly includes two branches: a detailed descriptor and a rough descriptor, and the two types of features obtained from the two branches are fused as the reconstructed features. The CSLR model based on this network is on the basis of the spatial-temporal hierarchical CSLR model MSTNet [21] proposed in our previous study, which first performs frame-level feature extraction on the source video data and then performs sparse sampling for the obtained frame-level features to reduce the computational effort of the frame-level feature extraction part of the spatial-temporal hierarchical model, followed by dense frame-level feature sequence reconstruction through the TSRNet, and finally uses CTC loss [22] training optimization after passing the temporal feature extraction part. The overall model is trained by the self-generating adversarial method proposed in this paper. The TSRNet is regarded as the generator, and the frame-level processing part and the time-series processing part are regarded as the discriminator. The training process is divided into two steps. In addition, this paper also proposes WERD as a new criterion for evaluating the effectiveness of the proposed network. The error rate between the estimated WER and the reference WER obtained, respectively, by the reconstructed frame-level feature sequence and the complete original frame-level feature sequence is taken as WERD.
The main three contributions of this paper are as follows:

- A TSRNet is proposed, which greatly reduces the computational complexity of the original spatial-temporal hierarchical CSLR model, and the network can be flexibly inserted into any spatial-temporal hierarchical CSLR model.
- The WERD is proposed as a new criterion to unify the criterion for model accuracy loss under different benchmarks.
- A self-generating adversarial training method is proposed to reduce the final error.

2 Related Work

This section will review related research from two aspects: existing CSLR methods, and related video super-resolution methods.

2.1 Continuous Sign Language Recognition

In the early days of CSLR, methods such as Hidden Markov Model (HMM) were usually used for recognition, but with the development of deep learning, CNNs were introduced into CSLR, and combined with HMM to form a “CNN+HMM” hybrid model [7,23–25]. With the emergence of RNN and CTC, the hybrid model composed of “CNN+RNN+CTC” using RNN instead of HMM is widely used in CSLR [1,26]. The hybrid model mainly uses 2D-CNN for frame-level feature extraction, then uses RNN for time-series processing and finally uses CTC for training and decoding. Huang et al. [27] developed a novel boundary-adaptive encoder-based approach for sign language recognition combined with window attention, achieving competitive results on popular benchmarks. Gao et al. [28] proposed an efficient RNN converter-based approach for Chinese sign language processing and designed a multi-level visual-level transcription network with frame-level, lexical-level and phrase-level BiLstm to explore multi-scale visual-semantic features. Min et al. [29] proposed visual alignment constraints to enable CSLR networks to be end-to-end trainable by enforcing feature extractors to predict with more alignment supervision to address the overfitting problem of CTC in sign language recognition. These methods can be classified as spatial-temporal hierarchical models and are the most widely used CSLR methods. In addition, some non-spatial-temporal hierarchical models such as “3D-CNN” and “2+1D-CNN” are also used in CSLR [30,31]. Ariesta et al. [32] proposed a sentence-level sign language method for deep learning combining 3D-CNN and Bi-RNN. Specifically, a 3D-CNN is used to extract features from each video frame, a Bi-RNN is used to extract unique features from the sequential behavior of video frames, and then a possible sentence is generated. Han et al. [8] used “2+1D-CNN” for feature extraction and proposed a lightweight spatial-temporal channel attention module, including two sub-modules of channel temporal attention and spatial-temporal attention, which enables the network to attend to important information in the spatial, temporal and channel dimensions by combining squeezed and elicited attention with self-attentiveness. This paper mainly conducts research on the basis of the spatial-temporal hierarchical model.

2.2 Video Super-Resolution

Video super-resolution is an extension of the image super-resolution task, which restores low-resolution video images into high-resolution video images [33]. It can fully utilize the inter-frame information in the restoration process to obtain better performance. Video super-resolution can be divided into two classes according to whether adjacent frames are aligned with the target frame: alignment methods and non-alignment methods [34]. Alignment methods include motion compensation methods and deformable convolution methods, and non-alignment methods include spatial non-alignment methods and spatial-temporal non-alignment methods [35–37]. These methods are all aimed at restoring the spatial resolution of the video. In video super-resolution, there is another method aimed at restoring the video timing resolution, which is video frame interpolation. This method is different from the method of restoring the video from low resolution to high resolution, in that it needs to make full use of the inter-frame information to restore the current frame in the time dimension [38]. Chen et al. [39] proposed an enhanced deformable separable network for video frame interpolation by processing adaptive offsets, kernels, masks, and biases learned from information in non-local neighborhoods, which has fewer parameters and improves the performance of kernel-based methods. Kalluri et al. [40] proposed a fully trainable process-free method for multi-frame video interpolation that aims to implicitly infer nonlinear motion trajectories and complex occlusions from unlabeled video, and greatly simplifies the process of training, testing and deploying frame interpolation models. In this study, our proposed method is most similar to the video frame interpolation method, which utilizes the inter-frame feature information to recover adjacent feature information in the temporal dimension using frame interpolation.

3 Methodology

The overall architecture of the CSLR model via TSRNet proposed in this paper is shown in Fig. 2. The model mainly consists of three parts: frame-level feature extraction, time-
series processing and TSRNet. In the model training stage, for the input sign language video, the frame-level feature sequence is first obtained through the frame-level feature extraction part. After down-sampling, a dense frame-level feature sequence is obtained through the TSRNet. Then, the final time-series features are obtained through the time-series processing part. Finally, the CTC loss is used for training optimization, and the sign language recognition results are obtained. The entire training phase is trained using our proposed self-generating adversarial training method, where the TSRNet is regarded as the generator, and the frame-level processing part and the temporal processing part are regarded as the discriminator. The network is trained using the down-sampled data of the frame-level feature sequence as the input of the TSRNet. In the model testing stage, the position of down-sampling is different from that in the training stage. At this time, the input sign language video is down-sampled directly and then the frame-level feature sequence is obtained through the frame-level feature extraction part. Subsequently, a dense frame-level feature sequence is obtained through the TSRNet. This is then passed through the time-series processing part. For the obtained temporal features, decoding is performed using CTC to obtain the final sign language recognition results.

The CSLR model via TSRNet is based on our previous research MSTNet [21]. The frame-level feature extraction part and the time-series feature extraction part of the model are consistent with MSTNet, that is, the frame-level features are based on resnet-34, and the time-series features are extracted using the “1DCNN+Transformer” coding structure. In order to further reduce the computational cost of the model and improve the real-time performance of the model, this paper proposes the TSRNet. The details of the TSRNet will be introduced in Sect. 3.1. The proposed specific training and testing methods will be introduced in detail in section C.

3.1 Temporal Super-Resolution Network

The TSRNet mainly includes a detail descriptor and a coarse descriptor, as shown in Fig. 3. The main branch is the detail descriptor. For the frame-level feature sequence with unified channel dimensions, the detailed description of the feature sequence is obtained by convolution of multiple 1-D residual blocks and 1-D transpose. Another branch is the rough descriptor, which directly up-samples the frame-level feature sequence to obtain a rough description of the feature sequence. The input of the TSRNet is obtained by sparsifying the frame-level feature sequence extracted from the frame-level feature extraction part of the spatial-temporal hierarchical model, and the output is the feature sequence obtained by fusing the detailed features and rough features. The feature sequence is the reconstructed dense frame-level feature sequence, which is used as the input of the sequential processing part of the subsequent spatial-temporal hierarchical model.

The TSRNet aims to resolve the degraded sparse frame-level feature sequences and thus recover the corresponding dense frame-level feature sequences. The whole super-resolution process can be understood as the inverse of the above-mentioned degradation process of dense frame-level feature sequences. For the dense frame-level feature sequence, the degradation process is as follows.
Fig. 3 Framework of temporal super-resolution network

\[
\hat{f} = \phi(f; \theta_a)
\]

(1)

where \( f \) denotes a dense frame-level feature sequence, \( \hat{f} \) denotes a sparse frame-level feature sequence, \( \phi(\cdot) \) denotes a degradation function, \( \theta_a \) denotes the parameters of the degradation function and various degradation factors, and in this paper only down-sampling methods are used.

Thus, the TSRNet can be expressed as \( \hat{f} = \phi^{-1}(\hat{f}; \theta_a) \), where \( \hat{f} \) denotes the estimate of the super-resolution of the frame-level feature sequence. The overall process of feature reconstruction via TSRNet is described in detail below.

For an input sign language video \( V = (x_1, x_2, \ldots, x_T) = \{x_t^T \in \mathbb{R}^{T \times h \times w} \} \) containing \( T \) frame, where \( x_t \) is the \( t \)-th frame image in the video, \( h \times w \) is the size of \( x_t \), and \( c \) is the number of channels. \( V \) passes through the frame-level feature extractor \( F_s \) of the spatial-temporal hierarchical model and obtains the feature expression as follows:

\[
\hat{f}_1 = F_s(V) \in \mathbb{R}^{T \times c_1}
\]

(2)

where \( c_1 \) is the number of channels after feature extraction. The dense frame-level feature sequence \( f_1 \) is down-sampled by \( n \) times to obtain a sparse frame-level feature sequence, and then the time dimension and the channel dimension are exchanged to obtain the feature sequence \( f_2 \in \mathbb{R}^{c_1 \times T_1} \), where \( T_1 = T/n \).

At this time, \( f_2 \) is the sparse feature sequence, which is the input of the TSRNet. Input \( f_2 \) to the two branches of the TSRNet, namely the detail descriptor and the coarse descriptor, respectively.

In the detail descriptor branch, the number of channels of the sparse feature sequence is first dimensionally increased by a 1D-CNN, and then batch normalized and activated by the activation function to obtain the sparse feature sequence \( f_3 \in \mathbb{R}^{c_2 \times T_1} \). The activation function \( \sigma \) is Relu, and \( c_2 \) is the number of channels after the ascension. Then the dimension-raising process can be described as:

\[
f_3 = \sigma(\text{BN}(1\text{-DCNN}(f_2))) \in \mathbb{R}^{c_2 \times T_1}
\]

(3)

Assuming that the entire dimension-raising process is \( F_{1\text{DCNN}}\)-Relu, Eq. (2) can be described as:

\[
f_3 = F_{1\text{DCNN}}\text{-Relu}(f_2) \in \mathbb{R}^{c_2 \times T_1}
\]

(4)

Then, for the sparse feature sequence \( f_3 \), the detailed features are extracted through several 1D-ResBlocks to obtain the sparse feature sequence \( f_3' \). The number of channels and the time length of \( f_3' \) remain unchanged, which is consistent with \( f_3 \).

\[
f_3' = F_{\text{Res}}(f_3) \in \mathbb{R}^{c_2 \times T_1}
\]

(5)

where \( F_{\text{Res}} \) indicates that the number of 1D-ResBlocks is \( m \).

After that, use 1-dimensional transposed convolution for the sparse feature sequence \( f_3' \) in the time dimension, and upsampling \( n \) times to obtain the dense feature sequence \( f_4 \in \mathbb{R}^{c_1 \times T} \), and the corresponding number of channels is also increased in dimension, \( c_3 \) is the number of channels after the up-dimension, \( T \) is the original video frame length. For \( f_4 \), the process of formula (4) is performed again, and the dense feature sequence \( f_4' = F_{\text{Res}}(f_4) \in \mathbb{R}^{c_3 \times T} \) is obtained by extracting features through \( k \) 1D-ResBlocks, and the number of channels and time length of \( f_4' \) remain unchanged.

\( f_4' \) is subjected to a 1D-CNN for dimensionality reduction, so that the number of channels is restored to be consistent with the input feature sequence, and then batch normalization is performed to obtain a dense feature sequence \( f_5 \).

\[
f_5 = \text{BN}(1\text{-DCNN}(f_4')) \in \mathbb{R}^{c_1 \times T}
\]

(6)

Assuming that the entire dimensionality reduction process is \( F_{1\text{DCNN}} \), Eq. (5) can be described as:

\[
f_5 = F_{1\text{DCNN}}(f_4') \in \mathbb{R}^{c_1 \times T}
\]

(7)

In the rough descriptor branch, only the nearest neighbor interpolation \( F_{\text{nearest}} \) is used for the sparse feature sequence
we use 1D depth-wise convolution in 1D-ResBlock, and its features into our model. Because we deal with 1D temporal data, the number of parameters is greatly reduced. This paper introduces the ResBlock structure of 1D-ResBlock with depth-wise convolution in ResBlock has no achieved excellent results. In the proposed model ConvNeXt, the performance, it was introduced into other related fields and was used in classification tasks. Due to its excellent performance, it was introduced into other related fields and achieved excellent results. In the proposed model ConvNeXt, Liu et al. [42] verified that the replacement of ordinary convolution with depth-wise convolution in ResBlock has no effect on model performance, but the amount of parameters is greatly reduced. This paper introduces the ResBlock structure into our model. Because we deal with 1D temporal data, we use 1D depth-wise convolution in 1D-ResBlock, and its structure is shown in Fig. 4.

Let the 1D-ResBlock map of layer be:
\[ f(x) = \sigma(BN(F_d(x)) + x) \] (10)
where \( x \) is the input data, \( y \) is the output data, and \( F_d(\cdot) \) is the 1D depth-wise convolution.

### 3.2 Connectionist Temporal Classification

CSLR belongs to weakly supervised learning. The input video is an unsegmented sequence and lacks a strict correspondence between video frames and labeled sequences. After encoding the input video sequence, it is very appropriate to use CTC as a decoder. CTC was originally designed for speech recognition, mainly to perform end-to-end temporal classification of unsegmented data to address the problem of mismatched lengths of input and output sequences. In recent years, it is often used in CSLR. In the CSLR model via TSRNet proposed in this paper: in the model training stage, for the final dense feature sequence, CTC Loss is used to train and optimize to obtain the optimal model; in the model testing stage, for the final time-series features, using CTC decoding to get the final sign language recognition result.

CTC introduces a blank label \([-\) to mark unclassified labels during decoding, that is, any words in the input video clip that do not belong to the sign language vocabulary, so that the input and output sequences can be matched, and the dynamic programming method is used for decoding [29].

For the input video \( V \) of \( T \) frames, the label of each frame is represented by \( \pi = (\pi_1, \pi_2, \ldots, \pi_T) \), where \( \pi_T \in V \cup \{\text{-}\} \), and \( V \) is the sign language vocabulary; then, the posterior probability of the label is:
\[
p(\pi | V) = \prod_{t=1}^{T} p(\pi_t | V) = \prod_{t=1}^{T} Y_{t, \pi_t} \] (11)

For a given sentence-level label \( s = (s_1, s_2, \ldots, s_L) \), where \( L \) is the number of words in the sentence. CTC defines a many-to-one mapping \( B \), whose operation is to remove blank labels and duplicate labels (e.g., \( B(-g-r-e-n-) = B(-g-r-e-n-) = \text{green} \)) in the path; then, the conditional probability of label \( s \) is the sum of the occurrence probabilities of all corresponding paths:
\[
p(s | V) = \sum_{\pi \in B^{-1}(s)} p(\pi | V) \] (12)
where \( B^{-1}(s) = \{ \pi | B(\pi) = s \} \) is the inverse mapping of \( B \). CTC loss is defined as the negative log-likelihood of the conditional probability of \( s \):
\[
L_{CTC} = -\ln p(s | V) \] (13)

The final sign language recognition result is obtained by CTC decoding after normalization by Softmax function.

### 3.3 Model Training and Testing Process

The TSRNet proposed in this paper belongs to the super-resolution model. The common training method in super-resolution models is to use L1 loss or L2 loss as the loss function, which reacts to the accuracy of the estimate by judging the distance between the estimate and the reference value, considering the difference at the individual data level [43,44]. However, what this paper wants to reconstruct is the...
feature vector. If only the gap between the data is considered, the result is bound to be unsatisfactory. Because each feature vector is a whole, it represents the integrated high-dimensional information, and the value may be very small after multiple feature extractions, so the similarity between vectors needs to be considered at this time. This paper draws on the training method of generative adversarial network (GAN) [45], applies it to our model training, and proposes a self-generating adversarial training method to train the TSR-Net, which greatly improves the final error rate. The training and testing process of the self-generating adversarial training method is described in detail below.

**Training** During the training process, we use the self-generating adversarial training method to train the temporal super-resolution network. We consider the temporal super-resolution network as the generator and the spatial-temporal hierarchical model as the discriminator. First, the original sign language video is input into the frame-level feature extraction part of the spatial-temporal hierarchical model to obtain the frame-level feature sequence, and the down-sampling data are used as the input of the TSRNet to train the network. And the training is divided into two steps, the first step is to train the spatial-temporal hierarchical model, and the second step is to train the TSRNet.

**Step 1** Train the spatial-temporal hierarchical model, as shown in Fig. 5. The original sign language video data are used as the input of CNN to extract frame-level features, and then the processed time-series features are obtained through the time-series processing module. Finally, the labeled video-level phrases are used as labels, and CTC Loss is used for training, and the final model obtained is used as the discriminator.

**Step 2** Train the TSRNet, as shown in Fig. 6. First, the TSRNet is inserted between the frame-level feature extraction part and the temporal feature extraction part of the spatial-temporal hierarchical model. The parameters of the spatial-temporal hierarchical model trained in the first step are frozen, and only the parameters in the TSRNet are trained. Then, the original sign language video is used as the input, and the frame-level feature sequence is obtained through the frame-level feature extraction part of the spatial-temporal hierarchical model. According to the set down-sampling multiple, the frame-level feature sequence is sparse. Then the sparse frame-level feature sequence is input into the TSRNet for reconstruction, and the dense frame-level feature sequence is reconstructed. Finally, the sequence is input into the temporal processing part of the spatial-temporal hierarchical model to obtain the final temporal features, and CTC loss is used for training according to the phrase annotation.

Here, in order to increase the robustness, proportional random sampling is used in the process of sparseness [46]. Assuming that the multiple of down-sampling is 4 and the length of the frame-level feature sequence is T, then the dense frame-level feature sequence slices the sequence with a width of 4 and divides it into n segments, where $n = T/4$. Then a random feature vector is taken from each segment and reconstructed into a sparse feature sequence, which is the input of the TSRNet.

**Testing** During testing, the difference from training is the location of down-sampling. Firstly, the input video data are directly down-sampled to obtain sparse video data, which is input to the frame-level extraction part of the spatial-temporal hierarchical model to obtain frame-level features. Then, a dense frame-level feature sequence is obtained by reconstruction through the TSRNet. The resulting dense frame-level...
The obtained timing features, the final recognition result is the component of the spatial-temporal hierarchical model. Finally, for the evaluation criteria proposed in this paper, we describe in detail in section C.

### 4.1 Dataset

**RWTH-PHONIX-Weather-2014 (RWTH) dataset** [47]: RWTH is recorded by a public weather radio station in Germany. All presenters wear dark clothing and performed sign language in front of a clean background. The videos in this dataset are recorded by 9 different presenters with a total of 6841 different signed sentences (where the number of sign language word instances is 77,321 and the number of vocabulary words is 1232). All videos are preprocessed to a resolution of 210×260, and a frame rate of 25 frames per second (FPS). The dataset is officially divided into 5672 training samples, 540 validation samples, and 629 test samples.

Chinese sign language (CSL) dataset [48]: CSL is captured using a Microsoft Kinect camera and contains 100 sentences of everyday Chinese language; each sentence demonstrates 5 times by 50 presenters with a vocabulary size of 178. The video resolution is 1280×720 and the frame rate is 30 FPS. The performance diversity of this dataset is richer because the demonstrators wear different clothes and demonstrate different speeds and ranges of motion. In the absence of official segmentation, we divide the CSL dataset into a training set and a test set according to the 8:2 rule. The training set accounts for 80% and the test set accounts for 20%, that is, it is divided into a training set of 20,000 samples and a test set of 5000 samples, and make sure that the sentences in the training and test sets are the same, but the presenters are different.

### 4.2 Implementation Rules

In the overall model of this paper, the Adam optimizer [49] is used for training, the initial learning rate and weight factor are set to $10^{-4}$, and the batch size used is 2. During model training, random cropping and random flipping are used for data augmentation. For random cropping, the frame size of each video sequence is first resized to 256×256 and then randomly cropped to a size of 224×224 to fit the shape of the input. For random flips, set flip probability to 0.5. Flip and crop processing is performed on video sequences. In addition, temporal enhancement processing is performed to randomly increase or shorten the length of the video sequence within ±20%. During model testing, only center cropping is used.
for data enhancement, and the beam search algorithm is used for decoding in the final CTC decoding stage, with a beam width of 10. For the RWTH dataset, there are 30 epochs in the training phase, and the learning rate is reduced by 80% at the 10th and 20th epochs. For the CSL dataset, there are 15 epochs in the training phase, and the learning rate is reduced by 80% at the 5th and 10th epochs. The graphics card used in this experiment is RTX2080Ti, the GPU dedicated memory size is 12 G, the CPU memory is 8 G, and the number of cores is 4.

### 4.3 Judgment Criteria

The WER is widely used as a criterion for CSLR [47]. It is the sum of the minimum insertion operations, replacement operations, and deletion operations required to convert the recognition sequence into a standard reference sequence. Lower WER means better recognition performance, which is defined as follows:

\[
WER = 100\% \times \frac{\text{ins} + \text{del} + \text{sub}}{\text{sum}}
\]  

(14)

where “ins” represents the number of words to be inserted, “del” represents the number of words to be deleted, “sub” represents the number of words to be replaced, and “sum” represents the total number of words in the label.

For the evaluation of the performance of the model in this paper, only using WER can represent the recognition performance of the model, but it cannot accurately represent the gap between the recognition results of the model in this paper and

| Down-sampling factor | TSRNet | Nearest neighbor interpolation | Linear interpolation |
|----------------------|--------|--------------------------------|----------------------|
|                      | WER (%)| WERD (%) | WER (%)| WERD (%) | WER (%)| WERD (%) |
| 1                    | 20.3   | 21.4    | 0      | 0       | 20.3   | 21.4    |
| 2                    | 20.7   | 21.5    | 1.9    | 0.5     | 21.8   | 22.3    |
| 3                    | 21.1   | 22.2    | 3.8    | 3.8     | 24.8   | 25.5    |
| 4                    | 23.4   | 24.7    | 14.7   | 15.6    | 26.3   | 27.4    |
| 5                    | 25.4   | 25.3    | 23.8   | 18.4    | 30.5   | 30.2    |
| 6                    | 28.2   | 28.9    | 36.0   | 34.3    | 33.5   | 33.9    |
| 7                    | 31.1   | 31.5    | 47.4   | 44.7    | 38.5   | 38.9    |
| 8                    | 35.3   | 34.9    | 61.4   | 56.7    | 43.8   | 43.4    | 80.8   | 78.1   |

| Down-sampling factor | TSRNet | Nearest neighbor interpolation | Linear interpolation |
|----------------------|--------|--------------------------------|----------------------|
|                      | WER (%)| WERD (%) | WER (%)| WERD (%) | WER (%)| WERD (%) |
| 1                    | 0.7    | 0       | 0.7    | 0       | 0.7    | 0       |
| 2                    | 0.7    | 0       | 0.8    | 0.5     | 0.8    | 0.5     |
| 3                    | 0.7    | 0       | 0.9    | 1.0     | 0.9    | 1.0     |
| 4                    | 0.8    | 0.5     | 1.3    | 2.9     | 1.4    | 3.3     |
| 5                    | 0.9    | 1.0     | 1.8    | 5.2     | 2.1    | 6.7     |
| 6                    | 1.1    | 1.9     | 2.6    | 9.0     | 3.1    | 11.4    |
| 7                    | 1.5    | 3.8     | 3.6    | 13.7    | 4.6    | 18.4    |
| 8                    | 1.9    | 5.7     | 5.4    | 22.0    | 6.9    | 28.7    |
| 9                    | 2.5    | 8.6     | 7.9    | 32.6    | 10.0   | 41.6    |
| 10                   | 3.8    | 14.7    | 11.1   | 45.9    | 13.6   | 54.7    |
| 11                   | 5.1    | 20.7    | 14.4   | 57.4    | 17.0   | 65.1    |
| 12                   | 6.9    | 28.7    | 17.6   | 66.7    | 21.3   | 75.4    |
| 13                   | 8.4    | 35.1    | 20.4   | 73.5    | 24.0   | 80.3    |
| 14                   | 10.3   | 42.8    | 24.0   | 80.3    | 28.3   | 86.6    |
the recognition results of the original model. Therefore, this paper proposes WERD to further measure the performance of this model. The frame-level feature sequence reconstructed by the model and the complete original frame-level feature sequence are both processed through the time-series processing part of the spatial-temporal hierarchical model and the final CTC decoding to obtain the final recognition result, and the WER is calculated to obtain the estimated WER and reference WER, respectively, and then the error rate between them is calculated as a further evaluation criterion for the model in this paper.

\[
\text{WERD} = 100\% \times \frac{1 - 1.1^{-(\text{WER}_E - \text{WER}_R)}}{1 + 1.1^{-(\text{WER}_E - \text{WER}_R)}}
\]  

(15)

where \(\text{WER}_E\) represents the estimated WER and \(\text{WER}_R\) represents the reference WER. Because \(\text{WER}_E\) is a process that approximates \(\text{WER}_R\), \(\text{WER}_E\) is greater than or equal to \(\text{WER}_R\). The smaller the WERD, the closer the estimated value is to the reference value, which means that the dense feature sequence reconstructed by the model is more similar to the original feature sequence. Ideally, \(\text{WERD} = 0\), that is, the estimated sequence is equal to the original sequence. When experimenting on the CSL dataset, we treat a single Chinese character as a word.

4.4 Experimental Results

In this paper, the model MSTNet in our previous study is used as the base model, and the TSRNet is inserted on its basis. The original data of the RWTH dataset and CSL dataset are down-sampled by different multiples as the input of the overall model, and the feature sequence is reconstructed using the proposed TSRNet. The experimental results are presented in Tables 1 and 2, respectively. The experimental results are compared with the other two methods for data recovery after down-sampling, and the comparison results are also presented in Tables 1 and 2. To further analyze the experimental data, the experimental results in Tables 1 and 2 are presented as graphs, as shown in Figs. 7 and 8.

From Tables 1 and 2, it can be obtained: Using different methods to reconstruct the down-sampled data under the same down-sampling factor, the final WERD obtained by our proposed temporal super-resolution network is smaller than the nearest neighbor interpolation method and the linear interpolation method, and when the down-sampling factor is larger, the advantage achieved by our method is more obvious.

For the RWTH dataset, the reference WER obtained on the validation set and test set, that is, the WER obtained without down-sampling the data, is 20.3% and 21.4%, respectively. When the down-sampling factor is 2, the WERD obtained by our method on the validation set and the test set is 1.9% and 0.5%, respectively, and the WERD using the nearest neighbor interpolation method is 7.0% and 4.3%, compared with our method, the error increased by 5.1% and 3.8%, respectively, the WERD using the linear interpolation method is 12.3% and 9.5%, and the error is increased by 10.4% and 9%, respectively, compared with our method. When the down-sampling factor is 3, the errors on the validation set and test set using the nearest neighbor interpolation method are increased by 17.3% and 15.5%, respectively. Compared with our method, the errors obtained by using the linear interpolation method are increased by 23.6% and 19.6%, respectively. When the down-sampling factor is 4, the errors on the validation set and the test set using the nearest neighbor interpolation method are increased by 13.1% and 12.2%, respectively, compared with our method, and compared with our method, the errors obtained by using the linear interpolation method are, respectively, increased by 26.5% and 24.4%. By analogy, the error is increasing.

For the CSL dataset, the reference WER for the test set is 0.7%. When the down-sampling multiples are 2, 3, 4 and
Table 3  Overall performance of continuous sign language recognition model via temporal super-resolution network

| Down-sampling factor | 1     | 2     | 3     | 4     | 5     | 6     | 7     | 8     |
|----------------------|-------|-------|-------|-------|-------|-------|-------|-------|
| **Frame-level feature extraction part** |       |       |       |       |       |       |       |       |
| Params               | $23 \times 10^6$ | $23 \times 10^6$ | $23 \times 10^6$ | $23 \times 10^6$ | $23 \times 10^6$ | $23 \times 10^6$ | $23 \times 10^6$ | $23 \times 10^6$ |
| FLOPs (GFlops)       | 734.74 | 367.37 | 242.47 | 183.69 | 149.95 | 121.23 | 102.87 | 91.84  |
| MemR + W (GByte)    | 14.94  | 7.47   | 4.93   | 3.73   | 2.99   | 2.46   | 2.09   | 1.87   |
| **Temporal super-resolution network** |       |       |       |       |       |       |       |       |
| Params               | $61 \times 10^6$ | $61 \times 10^6$ | $61 \times 10^6$ | $61 \times 10^6$ | $61 \times 10^6$ | $61 \times 10^6$ | $61 \times 10^6$ | $61 \times 10^6$ |
| FLOPs (GFlops)       | –      | 11.57  | 11.41  | 11.25  | 11.19  | 11.37  | 11.28  | 11.09  |
| MemR + W (GByte)    | –      | 0.36   | 0.35   | 0.34   | 0.34   | 0.34   | 0.34   | 0.33   |
| **Time-series feature extraction part** |       |       |       |       |       |       |       |       |
| Params               | $97 \times 10^6$ | $97 \times 10^6$ | $97 \times 10^6$ | $97 \times 10^6$ | $97 \times 10^6$ | $97 \times 10^6$ | $97 \times 10^6$ | $97 \times 10^6$ |
| FLOPs (GFlops)       | 1.79   | 1.79   | 1.79   | 1.79   | 1.79   | 1.79   | 1.79   | 1.79   |
| MemR + W (GByte)    | 0.035  | 0.035  | 0.035  | 0.035  | 0.035  | 0.035  | 0.035  | 0.035  |
| **Total** |       |       |       |       |       |       |       |       |
| Params               | $120 \times 10^6$ | $181 \times 10^6$ | $181 \times 10^6$ | $181 \times 10^6$ | $181 \times 10^6$ | $181 \times 10^6$ | $181 \times 10^6$ | $181 \times 10^6$ |
| FLOPs (GFlops)       | 736.53 | 380.73 | 253.88 | 196.73 | 159.93 | 134.39 | 115.94 | 104.72 |
| MemR + W (GByte)    | 14.94  | 7.87   | 5.31   | 4.11   | 3.36   | 2.83   | 2.46   | 2.24   |
| Params MEM (MB)      | 458.75 | 691.63 | 691.63 | 691.63 | 691.63 | 691.63 | 691.63 | 691.63 |
| Run time (ms)        | 221.57 | 137.71 | 100.78 | 73.5   | 58.74  | 52.08  | 48.56  |

5, the WERD obtained by our method on the test set is all
0, that is, there is no error. This is because the frame rate of
CSL is 30FPS/s, and the sign language demonstrator is slow
to demonstrate sign language, resulting in more redundancy
in the continuous sign language dataset. When the down-
sampling factor is 6, the WERD obtained by our method on
the test set is 0.5%, and the errors obtained by the nearest
neighbor interpolation method and the linear interpolation
method are increased by 2.4% and 2.8%, respectively, com-
pared with our method. When the down-sampling factor is 7,
the errors obtained by using the nearest neighbor interpola-
tion method and the linear interpolation method are increased
by 4.2% and 5.7%, respectively, compared with our method.

As can be seen from the curves in Figs. 7 and 8, as the
down-sampling factor increases, the WER obtained on the
two datasets using the three methods also increases. And the
WER obtained using our method TSRNet is much smaller
than the nearest neighbor interpolation method and the linear
interpolation method, proving the effectiveness of the TSR-
Net.

As can be seen from Fig. 7, for the RWTH dataset, the WER
curve of TSRNet increases abruptly after the down-sampling
multiple is 3, which indicates that the down-sampling mul-
tiple of 3 is an inflection point. From the calculation of the
experimental results in Table 1, it can be seen that when the
down-sampling multiple is 10, the WERD obtained on the
test set is 1.9% larger than that when the down-sampling mul-
tiple is 9; when the down-sampling factor is 11, the WERD
increases by 6.1% when the down-sampling factor is 11. This
is consistent with the conclusion that the down-sampling fac-
tor of 11 is an inflection point obtained in Fig. 8, which proves
that TSRNet has the best model performance when the down-sampling factor is 11 on the CSL dataset.

In order to further analyze the effectiveness and superi-
ority of TSRNet, under different down-sampling times, this
paper analyzes the overall model from three aspects: calcu-
lation amount, parameter amount and running time. The
original data, that is, the video frame length of 200 and the
image size of 224 × 224, are input into the overall model. The calculation amount, parameter amount and running time of the model under different down-sampling times are shown in Table 3. The running time is the average of five consecutive model running times.

As can be seen from Table 3, for the amount of parameters, the introduction of TSRNet on the basis of MSTNet results in an increase of $61 \times 10^6$ in the amount of parameters, and with the increase of the down-sampling multiple, the size of the parameter amount remains unchanged. As for the amount of computation, with the increase of the down-sampling multiple, the amount of computation keeps decreasing. And the model calculation amount is positively correlated with the running time. As the calculation amount decreases, the running time also decreases, as shown in Figs. 9 and 10. It can be seen from the previous experiments that TSRNet has the best model performance when the down-sampling multiple is 3 on the RWTH dataset. For Figs. 9 and 10, when the down-sampling multiple is 3, the overall calculation of the model is reduced by 65.53% and the running time was reduced by 54.52%.

Through the above experiments, it can be found that for the spatial-temporal hierarchical model, the calculation amount is mainly concentrated in the frame-level feature extraction part, and most of its parameters are concentrated in the time-series feature extraction part. The introduction of TSRNet greatly reduces the computational complexity of frame-level feature extraction, but increases the overall model parameters.

4.5 Ablation Experiment

In this section, we conduct ablation experiments on the RWTH dataset to further verify the effectiveness of each component of the model. WER and WERD are used as metrics in ablation experiments, and the down-sampling factor is set to 4.

1) Experiment 1 Overall model performance of inserting TSRNet into different spatial-temporal hierarchical models. The TSRNet proposed in this paper is inserted into different spatial-temporal hierarchical models as a sub-network to reduce the overall model computation and running time with an acceptable error rate. In this paper, TSRNet is inserted into 4 different spatial-temporal hierarchical models, and the WER and WERD obtained by the overall model are calculated, respectively, as shown in Table 4. The two models compared in Table 4 “CNN + BiLSTM + CTC” and “VAC” [29] are retrained by us.

It can be seen from Table 4 that under the same down-sampling factor, inserting TSRNet into different spatial-temporal hierarchical models obtains less variation in the WERD of the overall model. It means that TSRNet has good generalization in different spatial-temporal hierarchical models, so that the overall performance of the model remains stable.

2) Experiment 2 Overall model performance using different types of 1D-ResBlock. This paper introduces 1D-ResBlock in TSRNet, but the overall model performance obtained by using different types of 1D-ResBlock is different. This paper conducts experiments using 4 different types of 1D-ResBlock, respectively, and calculates the parameters of WER and TSRNet of the overall model under different types,
Table 4 Overall model performance comparison of inserting TSRNet into different spatial-temporal hierarchical models

| Down-sampling factor | CNN + BiLSTM + CTC | VAC [29] | MSTNet [21] |
|----------------------|--------------------|----------|-------------|
|                      | Dev Test           | Dev Test | Dev Test    | Dev Test | Dev Test | Dev Test |
| 1                    | 26.1 26.7          | 0 0      | 21.8 22.8   | 0 0      | 20.3 21.4| 0 0      |
| 4                    | 29.8 30.4          | 17.5 17.5| 25.1 26.2   | 15.6 16.1| 23.4 24.7| 14.7 15.6|

Table 5 Overall model performance comparison using different types of 1D-ResBlock

| Types of 1D-ResBlock | WER (%) | The amount of parameters of TSRNet |
|----------------------|---------|-----------------------------------|
|                      | Dev Test|                                   |
| A                    | 23.7 24.9| 350 × 10^6                        |
| B                    | 23.4 24.9| 230 × 10^6                        |
| C                    | 23.4 24.7| 61 × 10^6                         |
| D                    | 23.8 24.7| 61 × 10^6                         |

Fig. 11 Four different types of 1D-ResBlock

as shown in Table 5. The 4 different types of 1D-ResBlock used are shown in Fig. 11.

It can be obtained from Table 5 that the overall model with 1D-ResBlock type C has the best performance. When 1D-ResBlock types are C and D, the number of parameters is smaller than that when types are A and B, because types C and D use depth-wise convolution 1D-DCONV. For types C and D with the same amount of parameters, type C reduces the WER by 1.7% on the validation set compared to type D.

(3) Experiment 3 Overall model performance using different down-sampling methods. When down-sampling the data, this paper uses proportional random sampling. The overall model WER obtained when using different down-sampling methods is shown in Table 6.

It can be seen from Table 6 that when the proportional random sampling method is used, the WER obtained by the overall model is the lowest and the performance is the best. Compared with the equally spaced sampling method, the WER obtained on the test set is reduced by 1.2%, and the WER obtained by the random sampling method on the validation set and test set is reduced by 11.1% and 7.3%, respectively. Proportional sampling can ensure the integrity of video data as much as possible, and random sampling can make the overall model have better generalization.

(4) Experiment 4 Overall model performance using different training methods. The overall model in this paper is trained using our proposed self-generating adversarial training method. We compare the self-generating adversarial training method with the conventional super-resolution network training method, and the WER of the overall model is shown in Table 7. The conventional super-resolution network training method here refers to: down-sample the dense feature sequence and input it into TSRNet, use L2 Loss to train the reconstructed dense feature sequence and the reference dense feature sequence, and then insert the trained model into the spatial-temporal hierarchical model to obtain the WER of the overall model.

As can be seen from Table 7, the overall model WER obtained by using our proposed self-generating adversarial training method is reduced by 6.4% and 5.7% on the validation set and test set, respectively, compared with the conventional super-resolution network training method,
proving that the effectiveness of self-generating adversarial training methods. Using conventional super-resolution network training methods only considers gaps between data levels, while ignoring gaps at semantic levels. For CSLR, semantic-level information plays an extremely important role, and the semantic-level information can be better recovered using our proposed self-generating adversarial training method.

5 Conclusion

A deep learning-based spatial-temporal hierarchical CSLR model uses dense sampling when extracting information from raw videos. However, as the video length increases, the amount of computation increases exponentially, making the model unsuitable for processing long video data in practical applications, limiting the real-time application of the model. To address this problem, this paper proposes a new TSRNet and constructs a CSLR model based on TSRNet to reduce the computational complexity and improve the real-time performance of the CSLR model. The sparse frame-level features are passed through TSRNet to obtain a reconstructed dense sequence of frame-level features, followed by time-series feature extraction, and training and optimization using CTC loss. This paper also proposes a self-generating adversarial training method to train the overall model, which can better restore the semantic-level information and greatly reduce the model error. In addition, this paper proposes WERD as a new criterion to unify the criterion for model accuracy loss under different benchmarks. Experiments on two large-scale sign language datasets demonstrate the effectiveness of the proposed model, which greatly reduces the overall model computation load and greatly improves real-time performance under a certain range of accuracy loss. And TSRNet can be flexibly inserted into any spatial-temporal hierarchical model.

CSLR is designed to solve the communication problem between hearing-impaired people and normal people, and the computational load and memory footprint of the model need to meet real-time requirements. The network proposed in this paper improves the real-time performance of the original model, but there are two problems, that is, it only optimizes the calculation amount of the frame-level spatial feature extraction part and increases the memory footprint on the basis of the original model. And the model in this paper is only for the spatial-temporal hierarchical model, but not involved in the non-spatial-temporal hierarchical model. Therefore, how to design a more lightweight, more real-time and more generalized model is a problem worth studying.

Acknowledgements This work was supported in part by the Development Project of Ship Situational Intelligent Awareness System, China, under Grant MC-201920-X01, in part by the National Natural Science Foundation of China under Grant 61673129.

Data Availability The datasets used in the paper are cited properly.

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Wei, C.; Zhao, J.; Zhou, W.; Li, H.: Semantic boundary detection with reinforcement learning for continuous sign language recognition. IEEE Trans. Circuits Syst. Video Technol. 31(3), 1138–1149 (2020)
2. Papastratis, I.; Dimitropoulos, K.; Daras, P.: Continuous sign language recognition through a context-aware generative adversarial network. Sensors 21(7), 2437 (2021)
3. Rastgoo, R.; Kiani, K.; Escalera, S.: Sign language recognition: a deep survey. Expert Syst. Appl. 164, 113794 (2021)
4. Elakkiya, R.: Machine learning based sign language recognition: a review and its research frontier. J. Ambient Intell. Humaniz. Comput. 12(7), 7205–7224 (2021)
5. Sincan, O.M.; Keles, H.Y.: Using motion history images with 3D convolutional networks in isolated sign language recognition. IEEE Access 10, 18 608-18 618 (2022)
6. Zhou, H.; Zhou, W.; Zhou, Y.; Li, H.: Spatial-temporal multi-cue network for continuous sign language recognition. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34(07), pp. 13009-13 016 (2020)
7. Koller, O.; Zargarani, S.; Ney, H.: Re-sign: Re-aligned end-to-end sequence modelling with deep recurrent CNN-HMMS. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4297–4305 (2017)
8. Han, X.; Lu, F.; Yin, J.; Tian, G.; Liu, J.: Sign language recognition based on R(2+1)D with spatial-temporal-channel attention. IEEE Trans. Hum. Mach. Syst. 57, 687–698 (2022)
9. Khedkar, V.N.; Dr, S.K.; Prasad, A.; Mishra, A.; Saha, V.; Kumar, V.: Analysis of recent trends in continuous sign language recognition using NLP. Lib. Philos. Pract. 5231, 1522 (2021)
10. Adaloglou, N.; Chatzis, T.; Papastratis, I.; Stergioulas, A.; Papadopoulos, G.T.; Zacharopoulou, V.; Xydomoulis, G.I.; Atzakas, K.; Papazachariou, D.; Daras, P.: A comprehensive study on deep learning-based methods for sign language recognition. IEEE Trans. Multimed. 24, 1750–1762 (2021)
11. Du, Y.; Xie, P.; Wang, M.; Hu, X.; Zhao, Z.; Liu, J.: Full transformer network with masking future for word-level sign language recognition. Neurocomputing 500, 115–123 (2022)
12. Krizhevsky, A.; Sutskever, I.; Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Adv. Neural Inf. Process. Syst. 25, 84–90 (2012)
13. Cui, R.; Liu, H.; Zhang, C.: A deep neural framework for continuous sign language recognition by iterative training. IEEE Trans. Multimed. 21(7), 1880–1891 (2019)
14. De Coster, M.; Van Herreweghe, M.; Dambre, J.: Sign language recognition with transformer networks. In: 12th International Conference on Language Resources and Evaluation, pp. 6018–6024. European Language Resources Association (ELRA) (2020)
