Hindawi
Mobile Information Systems
Volume 2022, Article ID 8931035, 9 pages
https://doi.org/10.1155/2022/8931035

Research Article

A Video Key Frame Extraction Method Based on Multiview Fusion

Ken Chen,[1,2] Jun Wang,[1,2] Yang Yang,[1,2] Yong Tang,[3] Yong Zhou,[1] and Jin Zhu[3]

[1]*Sichuan Digital Transportation Tech Co.,Ltd, Chengdu, China*
[2]*Institute of Future Transportation Engineering, Chengdu, China*
[3]*Shudao Investment Group Co.,Ltd, Chengdu, China*

Correspondence should be addressed to Jun Wang; jwang203915@163.com

Received 21 March 2022; Revised 23 May 2022; Accepted 11 June 2022; Published 22 July 2022

Academic Editor: Sebastian Podda

A massive amount of video data is stored in the real-time road monitoring system, especially in high-speed scenes. Traditional methods of video key frame extraction have the problems of large computation and long-time consumption. Thus, it is imperative to decrease the massive video data generated by monitoring and help researchers to study key frames. Aiming at the above problems, we propose an efficient key frame extraction method based on multiview fusion, where the autoencoder is used to compress the video data. Specifically, all the video frames of the video data are subjected to feature dimensionality reduction, and the features after dimensionality reduction are subjected to multiview fusion. Finally, dynamic programming and clustering are used to extract key frames. The experimental results show that the proposed method has lower computational complexity in extracting key frames, while the mutual information in the extracted key frames is large. It illustrates the reliability and efficiency of the proposed method, which provides technical support for subsequent video research.

1. Introduction

With the development of the economy and the urgent needs of public travel, intelligent road and traffic monitoring devices have been increasingly adopted, including speed dome cameras, bayonet cameras, box cameras, and fisheye lenses. Also, the resolution and fidelity of videos have recently made significant progress (such as 4k, 8k, and 1 million pixels [1], high dynamic range [2], and bit depth [3]). Accordingly, a massive amount of video data is collected from those various video acquisition devices, which has been a huge burden on the transmission and storage of video data. For large-scale video data, it is impossible to save all video frames in the storage system, which will cause huge memory consumption. Therefore, it is necessary to extract key frames of video data, which can represent the scene content [4].

In the transportation industry, real-time video surveillance and large-scale video segment processing have brought huge challenges to key frame extraction. In terms of time constraints, for monitoring purposes, we need real-time processing speed to meet the frame rate of video cameras from 24 to 30 frames per second (FPS) [5]. In terms of resource constraints, firstly, key frame extraction requires a lot of computing resources, particularly when each frame needs to be processed. Secondly, the continuous transmission of video streams adds a huge burden to network traffic and requires high network bandwidth.

In order to quickly access useful video information from a large amount of video data, the key frame extraction technology has been proposed. In recent years, the research on key frame extraction has made some achievements. Nasrollahi et al. [6] introduced four indicators such as symmetry, sharpness, contrast, and brightness to evaluate the image quality used for key frame selection. Anantharajah et al. [7] applied a similar metrics-based quality assessment system to face clustering in news videos. However, all of the above work requires predefined empirical weights to be associated with different quality measures in order to form the final quality score. However, fixed weights are difficult to adapt to different videos in different scenes. Luo et al. [8] used moving object detection and image similarity to extract key frames from surveillance video, but the calculation speed is slow due to the extraction of key frames for global features, which is not conducive to large-scale surveillance data. Hyesung et al. [9]

proposed a deep learning semantic-based scene-segmentation model that considers image captioning to segment a video into scenes semantically, but this method does not apply to high-speed scenes. Carta Salvatore et al. [10] proposed VSTAR (Visual Semantic Thumbnails and tAgs Revitalization), a novel approach in video optimization that exploited image captioning to simultaneously suggest tags and thumbnails.

In recent years, deep learning [11, 12] has shown great advances in image recognition, face recognition, and other related fields [13]. In addition, deep learning has also achieved great success in the field of video images. More and more researchers tend to use deep learning methods [14].

In order to solve the problems of the existing methods, we propose a global key frame extraction method based on an autoencoder. Firstly, we use part of the video frame data to train the network model. Then, we use the trained model to reduce the dimension of all the video frames and perform multiview fusion for low-dimensional features [15]. Finally, we use the fused features to extract key frames.

The main contributions of the paper are as follows:

(i) By combining information from multiple views, we can extract key frames more accurately

(ii) Feature extraction is performed on multiple video frames simultaneously, thereby significantly reducing redundant information of video frames

(iii) We propose a new loss function to ensure the consistency of image data before and after dimensionality reduction

(iv) We use low-dimensional features for key frame extraction, which will improve the speed of key frame extraction

The rest of this article is organized as follows. The second part introduces the content of the autoencoder. The third part introduces the multiview feature dimension reduction model of the video stream. The fourth part introduces multiview fusion key frame extraction and gives experimental results. The fifth part is the conclusion of this article.

2. Reviews of the Autoencoder

In 1986, Rumelhart proposed the concept of autoencoder and applied it to high-dimensional complex data processing, thus promoting the development of neural networks [16]. Autoencoder is a feedforward noncyclic neural network, an unsupervised neural network model, which can learn the hidden features of the input data. It has a good ability to extract data features and is also an important part of a deep trust network. It has a wide range of applications in the fields of image reconstruction, clustering, and machine translation. The autoencoder is applied to video key frame extraction, and the implicit features are learned by dimensionality reduction of video data features. Finally, the hidden features of the video are used for key frame extraction.

2.1. Network Structure. The basic structure of the autoencoder is shown in Figure 1, including encoding and decoding.

The autoencoder encodes the input x to obtain new features and expects that the original input can be reconstructed from the new features. The encoding process is as shown under the formula as follows:

$$y = f(Wx + b). \tag{1}$$

It can be seen that, just like the structure of a neural network, its coding is a linear combination, followed by a nonlinear activation function. If there is no nonlinear packaging, there is no difference between the autoencoder and ordinary PCA. After encoding the data, use the new feature y to reconstruct the input x, that is, the decoding process; the decoding process formula is as follows:

$$x' = f(W'y + b'). \tag{2}$$

In order to make the reconstructed x' and the input x as consistent as possible, a variety of loss functions can be used to iteratively train the network model.

2.2. Network Training Process. Similar to the deep feedforward network, the training process of the autoencoder network mainly consists of parameter initialization, forward propagation calculation, and error backward-propagation updating weight. The specific training process is shown in Figure 2.

3. Video Stream Multiview Feature Dimensionality Reduction Model

3.1. Data Acquisition and Preprocessing. Real-time video stream data were obtained by a bayonet camera of an operational highway in Sichuan Province. The total length of the obtained video stream is 4 minutes, and the video data are decomposed into video frames with a total of 6000 video frames. After obtaining the video frame data required by the training network model, due to the format of the data, further processing of the data is needed, so that the data can be used as the training set of deep neural network to train the network model. The specific processing process is as follows.

In the first step, we use the ViBe algorithm fusing the interframe difference method to divide the original video into 10 segments containing the moving object and select 300 video frames from each segment as training data.

The second step is to convert the video frame into RGB three-channel data on the computing plane. The RGB three-channel data can be directly expressed as three matrices. Therefore, we can easily sample fixed-size data on the calculation plane.

The third step is to normalize the RGB three-channel data of the key frame. The data normalization method we used was z-score-sigmoid normalization [17], which first performed z-score normalization of the data and then signed normalization.

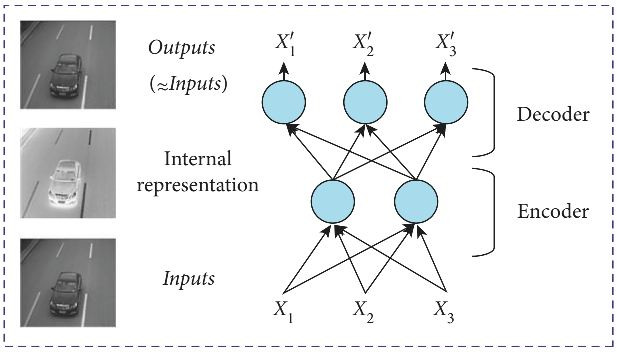

FIGURE 1: Encoding and decoding of autoencoder.

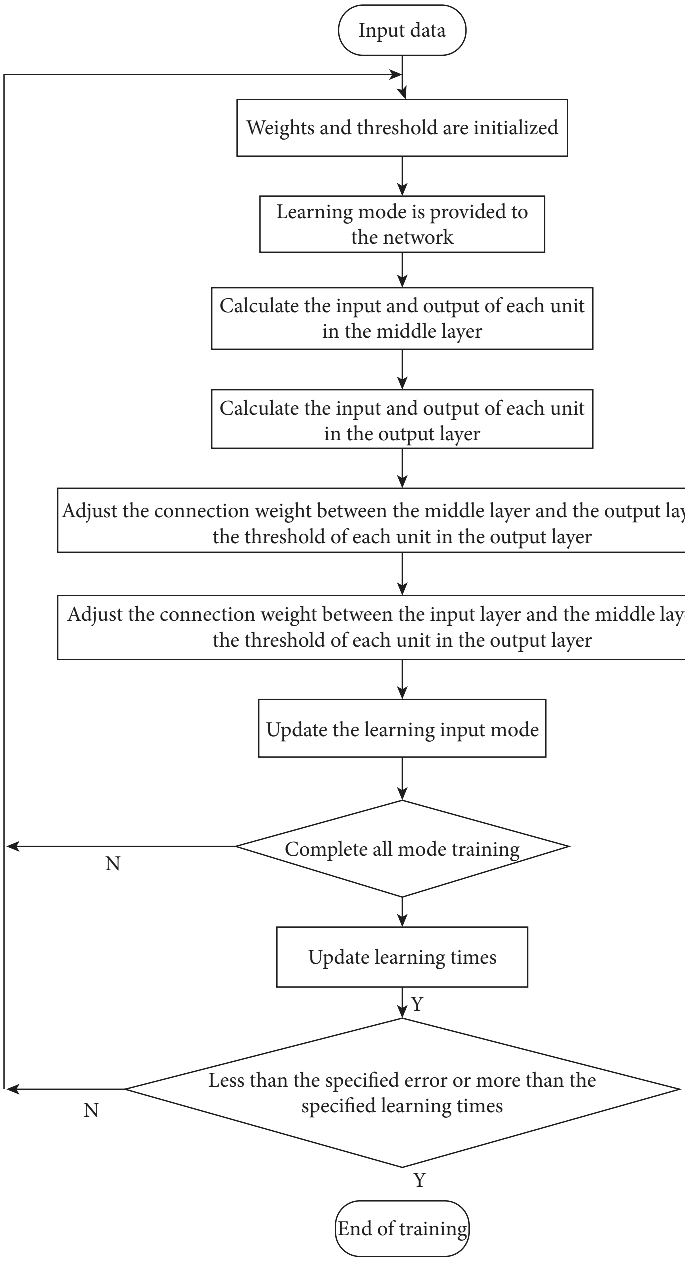
Input data
Weights and threshold are initialized
Learning mode is provided to the network
Calculate the input and output of each unit in the middle layer
Calculate the input and output of each unit in the output layer
Adjust the connection weight between the middle layer and the output layer and the threshold of each unit in the output layer
Adjust the connection weight between the input layer and the middle layer and the threshold of each unit in the output layer
Update the learning input mode
Complete all mode training
N
Update learning times
Y
Less than the specified error or more than the specified learning times
N
Y
End of training

FIGURE 2: Overall flow chart of network training.

$$X = \frac{1}{1 + e^{-(x-\mu/\sigma)}}. \tag{3}$$

In the last step, the normalized data of R, G, and B channels obtained above are converted into the data format required by the training network model.

3.2. Network Model Design. After the preprocessing part, the autoencoder network model used includes encoding and decoding. The encoder network model in the autoencoder includes one input layer, six convolution layers, and two full connection layers. The convolutional layer is shown in Figure 3. From Conv1 to Conv6, the number of feature graphs is 8, 16, 32, 64, 128, and 128, respectively. All these convolution layers use 3 × 3 convolution kernel. The output of the last convolutional layer is input to the fully connected layer, and an output of 2560 is obtained. The output after the fully connected layer is a 1024 vector. The decoder network model part in the autoencoder consists of a fully connected layer, a reshape layer, and six deconvolution layers. The number of neurons in the fully connected layer is 1280. The size of the six deconvolution layers is 5, 4, and 128 as shown in Figure 3. The number of characteristic diagrams from DeConv1 to DeConv6 is 128, 64, 32, 16, 8, and 2, respectively. The activation function used in the convolution layer and the deconvolution layer is the linear rectifying function (ReLU) [18]. The overall network model diagram is shown in Figure 3.

A specific loss function is set for the network model, and then the optimal network model is obtained by iterative training according to the loss function. In the process of training, the network model needs to be iterated repeatedly to continuously reduce the difference between the prediction and the real results. The loss function needs to be used. The average absolute error loss function is taken as part of the loss function of the network model. MAE is defined as the following formula, where y represents the true result and h represents the predicted output of the network model:

$$\mathrm{MAE}(X, h) = \frac{1}{m}\sum_{i=1}^{m}\left|h(x_i) - y_i\right|. \tag{4}$$

The loss function used in this paper is the mixed loss function, and the specific form is shown in the following formula:

$$\mathrm{loss}_{\mathrm{net}} = \beta \mathrm{loss}_{\mathrm{MAE}} + (1-\beta)\mathrm{loss}_{\mathrm{gradient}}, \tag{5}$$

where β is determined randomly and the value of β is adjusted according to the result of reconstruction $(0 \le \beta \le 1)$. $\mathrm{loss}_{\mathrm{gradient}}$ refers to the addition of gradient information between each line of video frame data. Its purpose is to ensure physical consistency between data after dimensionality reduction and data before dimensionality reduction, so that the restored data are consistent with the gradient of the original data. Formula (6) expresses that each point is determined by the difference between its upper, lower, left, and right four points, where x_{ij} represents the variable after reconstruction from the encoder. The calculation of $\mathrm{loss}_{\mathrm{gradient}}$ is shown in formula (7).

$$H(X) = \frac{1}{4}\left[\left|x_{i,j-1} - x_{i,j}\right| + \left|x_{i-1,j} - x_{i,j}\right| + \left|x_{i+1,j} - x_{i,j}\right| + \left|x_{i,j+1} - x_{i,j}\right|\right], \tag{6}$$

$$\mathrm{loss}_{\mathrm{gradient}} = \frac{1}{NM}\sum_{N-1}^{N-1}\sum_{j=0}^{M-1} H\left(x_{ij}\right). \tag{7}$$

The optimization algorithm for network training selects Adam [19] algorithm, where the step size is set to 0.001, the parameter β_1 is set to 0.9, β_2 is set to 0.99, and the initial value of the learning rate is set to 0.001; the number of batch samples is set to 128, that is, each input 128 time steps of video data for training; the maximum number of iterations for training is set to 5000.

Train the network model according to the acquired training data and the corresponding training strategy. The training process of the network is described as follows.

The first step is to input the training data into the autoencoder network model and initialize the network model parameters.

The second step is to perform the forward calculation of the network model, calculate the initial loss function of the network model, and use the optimization algorithm to optimize the network model parameters.

The third step is to use the backpropagation algorithm to optimize the network model and adjust the network model parameters.

The fourth step is to repeat the above second and third steps and iteratively train until the loss function stabilizes or is less than the specified threshold.

4. MultiView Fusion Key Frame Extraction

We use the low-dimensional features encoded by the convolutional autoencoder for video key frame extraction. The specific implementation process is shown in Figure 4. Firstly, the video frame is feature reduced, and then the features after the dimensionality reduction are used for key frame extraction. The dynamic programming method and the clustering method to are used extract the video key frames, respectively.

4.1. Video Frame Reconstruction. In order to compare the effect of the mean absolute error (MAE) and the mixed loss function training network model, the network model with two loss functions both attaining the minimum was obtained by training. Data of three key frames are selected for comparison of loss functions, as shown in Figure 5.

By comparing the results before and after reconstruction, it can be obtained that the network model using the mixed loss function has the best recovery effect on the video frame. Compared with the average absolute error loss function, the mixed loss function reduces the noise of the reconstruction result and can be better. The restored video frame is closer to

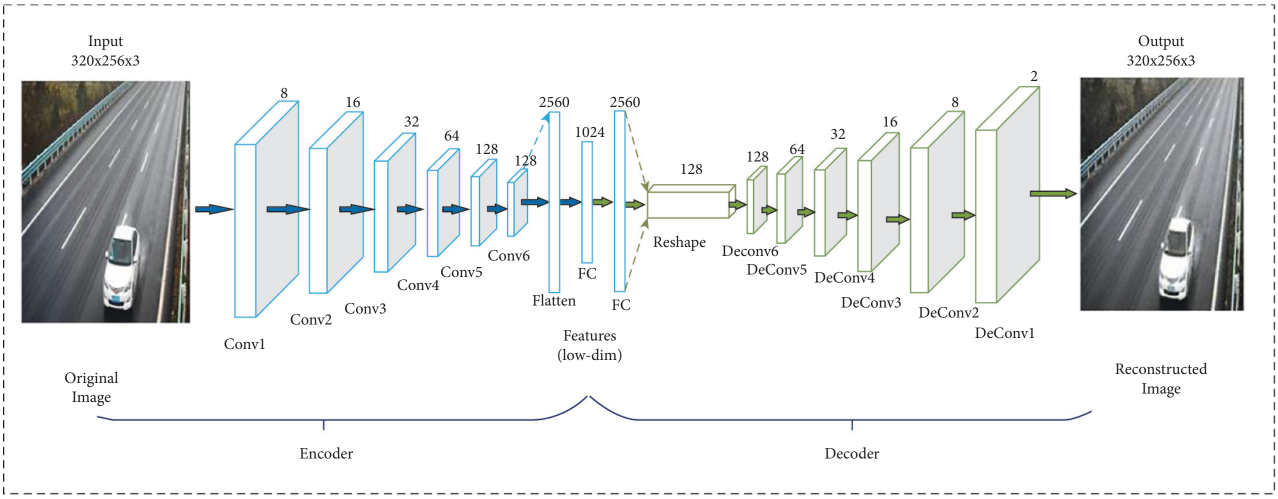

Figure 3: Convolutional autoencoder network structure.

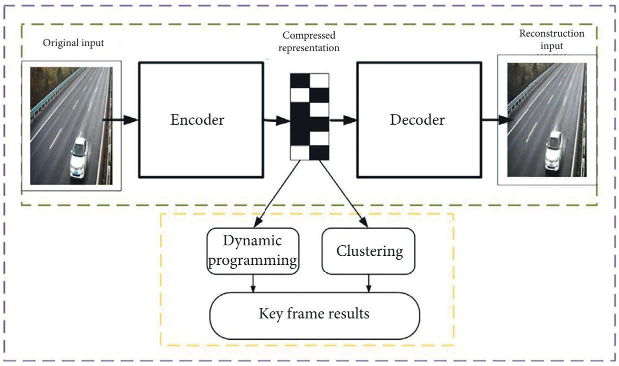

Figure 4: Key frame extraction algorithm process.

the original video frame. Therefore, the network model trained with the mixed loss function has a better reconstruction effect on the data. For feature dimensionality reduction, the hybrid loss function proposed in this paper is used.

Three key frames of data to refactor the test network model are white car data (44–49 frames), red truck data (928–933 frames), and multiple vehicle data (2937–2942 frames). We perform feature dimension reduction on the selected video frame data, reduce high-dimensional features of low-dimensional features, and finally use low-dimensional features to reconstruct video frames. The results of comparing the video frame data before and after reconstruction are shown in Figure 6.

Comparing the results before and after reconstruction, it can be seen that the network model we proposed can well extract the features of different scenes, and the low-dimensional features after dimensionality reduction can be well reconstructed back to the original high-dimensional features.

We reconstructed the video frames of single cars, trucks, and multiple vehicles. It can be seen from Figure 6 that the autoencoder network model can reconstruct high-dimensional features using low-dimensional features, and the network model has good generalization performance, so the low-dimensional features after dimensionality reduction are used for key frame extraction.

4.2. Multiview Fusion Key Frame Extraction. The key frame extraction of multiview fusion proposed in this paper is characterized by using the three channels of image data R, G, and B for feature reduction and then performing a multiview fusion of the data of the three channels after the dimensions. The fusion method adopted is that the three channels are directly spliced, and finally the key frames are extracted using dynamic programming and clustering methods, respectively.

The dynamic programming algorithm, the clustering algorithm, and our method are implemented in Python. All

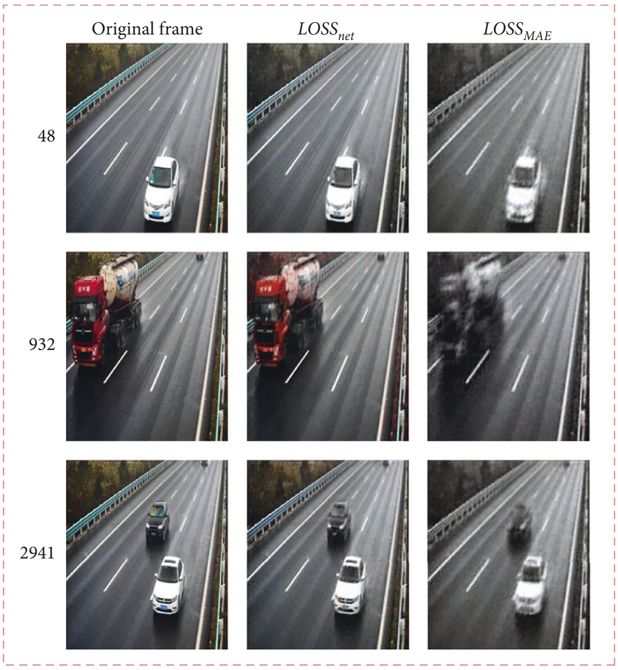

Figure 5: Comparison of loss_{MAE} loss function and hybrid loss function loss_{net}.

methods are run on the same desktop computer, which has an Intel (*R*) Xeon (*R*) Gold 6144 CPU @ 3.50 GHz and 64 GB of memory.

The concept of dynamic programming is used to extract video key frames. The specific step is to discover the shortest path of the sum of distances from the first video frame to the last video frame from multiple time steps. When the sum of distances is the slightest, it means that the preserved videos are all videos of key frames. In the process of key frame extraction by dynamic programming, there are many different distance functions to choose from. The distance functions we used include Mutual Information (MI), Euclidean Distance (ED), and Mahalanobis Distance (MD).

In order to verify the effectiveness of the algorithm, the video frames without feature dimensionality reduction and feature dimensionality reduction are used, and the key frames are extracted by dynamic programming method and clustering method to verify the effectiveness of the algorithm. In the key frame extraction experiment, 10 key frame data are extracted from 4-minute video data. The comparison between the video data for feature dimensionality reduction and the original video data is shown in Table 1.

After processing the video data through the above process, the storage space of the video data is reduced by 71.43%, the video frame after dimensionality reduction is reduced by 98.13%, and the key frame extraction using the dimensionality reduced data reduces the required computing resources and time consumed.

MI indicates that the distance function used for dynamic programming is mutual information, ED indicates that the distance function used is Euclidean distance, MD indicates that the distance function used is Mahalanobis distance, and AE + MI indicates Auto Encode plus MI, K-medoids clustering method, and AE + K-medoids. Compared with the method of combining AE and dynamic programming for feature extraction and direct dynamic programming, the method that uses AE consumes less time for key frame extraction, and the total mutual information is greater (the total mutual information is obtained by calculating the sum of the mutual information between adjacent time steps of all extraction results.) The larger the mutual information, the smaller the correlation between key frames and the more accurate the extraction result.

The results in Table 2 show that the method that uses AE is compared with the method that does not use AE. The method that uses AE has greater total mutual information and takes less time to extract key frames, and the extracted key frames are evenly distributed. Therefore, the result of using the self-encoding method for key frame extraction is more accurate and more representative. According to the results of Figure 7, using the dynamic programming method will extract a partially repeated background image, which is not representative, while the method using the encoder and dynamic programming can solve the above problems well. The difference between each video frame extracted by AE + MI is large, which can better replace the change of the whole video.

As shown in Table 2 and Figure 8, the total mutual information of AE + K-medoids is larger than that of K-medoids only, demonstrating that the method using AE + K-medoids can select more representative results from multiple key frames and consumes less time.

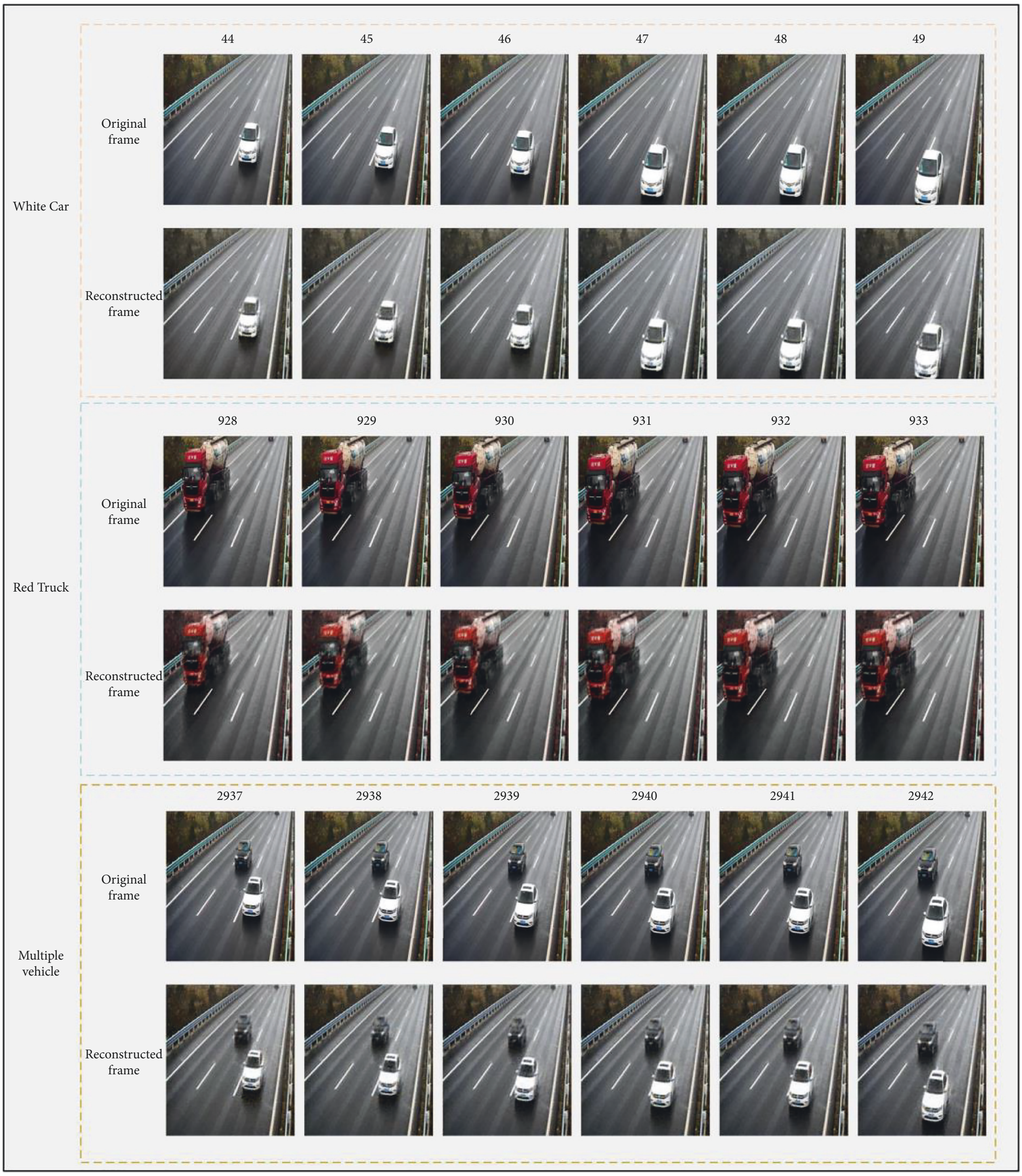

Figure 6: Comparison picture of vehicle reconstruction before and after different video frames.

The above empirical results demonstrate that the method of self-encoder plus dynamic programming based on mutual information and self-encoder plus clustering has a better effect on key frame lifting than other methods. The extracted key frames can represent the changes in video features, and the extracted key frames are evenly distributed. Compared with direct key frame extraction, the method proposed in this paper improves the accuracy of key frame results to a certain extent and reduces the time of key frame extraction.

Table 1: Comparison table before and after dimensionality reduction of video data features.

Data	Number of original video frames	Storage space required (GB)	Key frame extraction input data	Storage reduces redundancy (%)	Input reduces redundancy (%)
Original data	6000	1.75	6000 × 320 × 256	**71.43**	**98.13**
Processed data	**3000**	**0.5**	**3000 × 3 × 1024**		

Table 2: Comparison table of total mutual information and running time of key frame video.

Data	MI	AE + MI	ED	AE + ED	MD	AE + MD	K-medoids	AE + K-medoids
Total-MI	2.966	**10.289**	2.493	7.004	2.718	7.912	2.3082	**10.340**
Time (s)	102.9	**10.02**	103.7	11.52	104.6	11.22	100.43	**10.13**

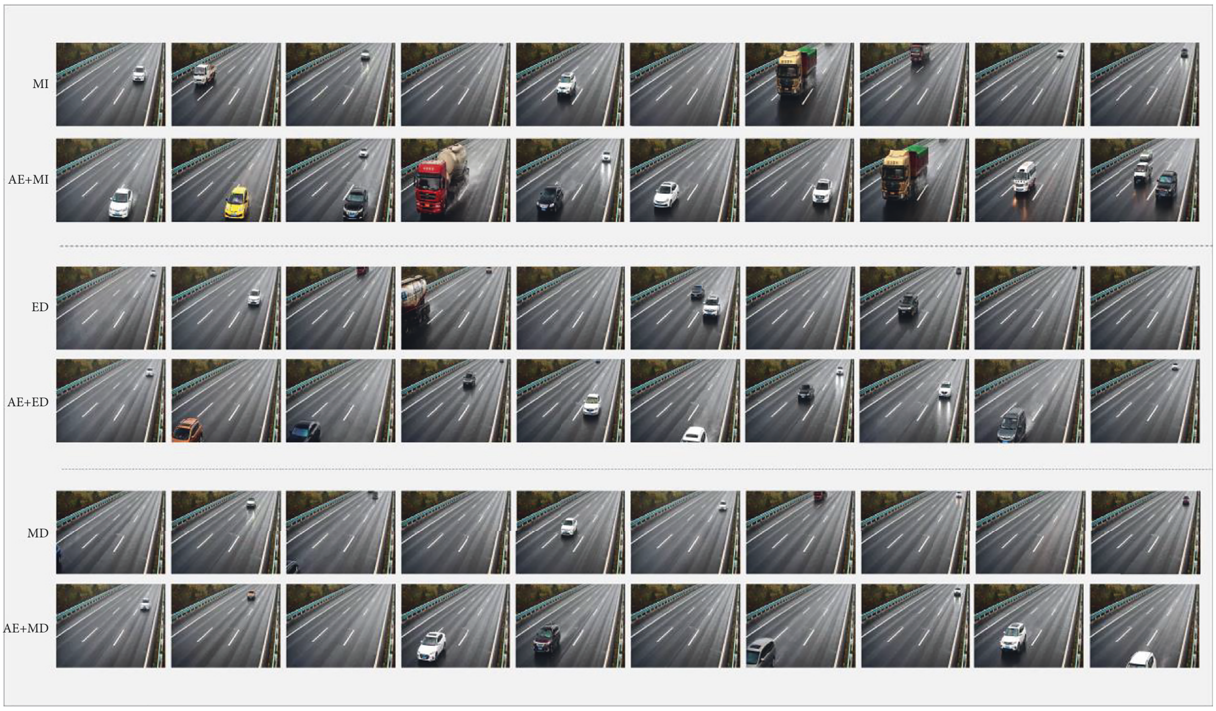

Figure 7: Dynamic programming key frame extraction results.

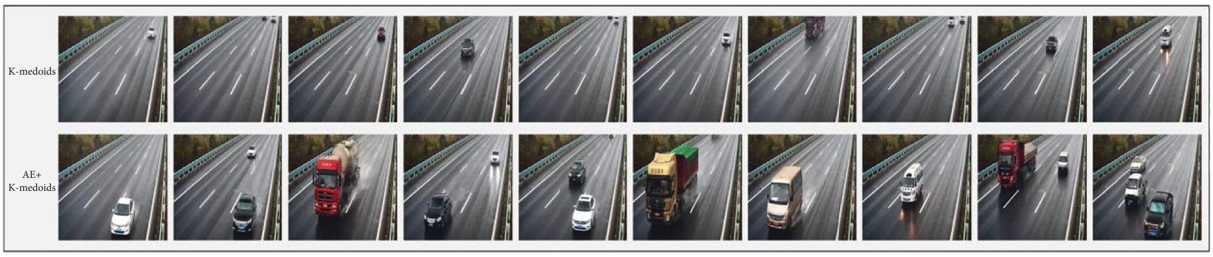

Figure 8: Clustering key frame extraction results.

5. Conclusion

We propose a key frame extraction method for video stream with multiview fusion. Firstly, video frames are used to train the autoencoder network model. Secondly, the network model is used to reduce the feature dimension of video frames, converting high-dimensional features into low-dimensional features, and then the multiple views after dimensionality reduction are fused. Finally, dynamic programming and clustering are used to extract key frames from the features of focused multiviews. The advantage of the above process for video key frame extraction lies in the dimensionality reduction of video data by using an autoencoder, which reduces a lot of computing time and resource consumption. Compared with the existing methods, the method proposed in this paper can extract the video data key frames more accurately, which is of great help to solve the key frame extraction of large-scale video data. However, the proposed method still requires the input video frame to be compressed or expanded to a specified size. Thus, future works will explore a simpler and faster network model for key frame extraction without limiting the size of video frames.

Data Availability

The data that support the findings of this study are available on request from the corresponding author.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work of this article was supported by Sichuan Digital Transportation Tech Co., Ltd and Institute of Future Transportation Engineering.

References

[1] D. J. Brady, M. E. Gehm, R. A. Stack et al., “Multiscale gigapixel photography,” *Nature*, vol. 486, no. 7403, pp. 386–389, 2012.

[2] S. B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, “High dynamic range video,” *ACM Transactions on Graphics*, vol. 22, no. 3, pp. 319–325, 2003.

[3] M. Winken, D. Marpe, H. Schwarz, and T. Wiegand, “Bit-depth scalable video coding,” in *Proceedings of the 2007 IEEE international conference on image processing*, IEEE, San Antonio, TX, USA, September 2007.

[4] Y. Zhuang, Y. Rui, T. S. Huang, and S. Mehrotra, “Adaptive key frame extraction using unsupervised clustering,” *IEEE* in *Proceedings of the 1998 international conference on image processing*, vol. 1, pp. 866–870, Chicago, IL, USA, October 1998.

[5] X. Qi, C. Liu, and S. Schuckers, “CNN based key frame extraction for face in video recognition,” in *Proceedings of the 2018 IEEE 4th International Conference on Identity, Security, and Behavior Analysis (ISBA)*, pp. 1–8, IEEE, Singapore, January 2018.

[6] K. Nasrollahi and T. B. Moeslund, “Face quality assessment system in video sequences,” in *Proceedings of the European Workshop on Biometrics and Identity Management*, pp. 10–18, Springer, Berlin, Heidelberg, May 2008.

[7] K. Anantharajah, S. Denman, and D. Tjondronegoro, “Quality based frame selection for face clustering in news video,” in *Proceedings of the 2013 international conference on digital image computing: techniques and applications (DICTA)*, pp. 1–8, IEEE, Hobart, TAS, Australia, November 2013.

[8] Y. Luo, H. Zhou, Q. Tan, Chen, and Yun, “Key frame extraction of surveillance video based on moving object detection and image similarity,” *Pattern Recognition and Image Analysis*, vol. 28, no. 2, pp. 225–231, 2018.

[9] H. Ji, D. Hooshyar, K. Kim, and Lim, “A semantic-based video scene segmentation using a deep neural network,” *Journal of Information Science*, vol. 45, no. 6, pp. 833–844, 2019.

[10] S. Carta, A. Giuliani, L. Piano, A. S. Podda, and D. R. Recupero, “VSTAR: visual semantic thumbnails and tAgs revitalization,” *Expert Systems with Applications*, vol. 1, Article ID 116375, 2022.

[11] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[12] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*, MIT press, Cambridge, MA, USA, 2016.

[13] Y. Bengio, *Learning Deep Architectures for AI*, Now Publishers Inc, Delft, Netherlands, 2009.

[14] J. Schmidhuber, “Deep learning in neural networks: an overview,” *Neural Networks*, vol. 61, pp. 85–117, 2015.

[15] M. Liang, Y. Zhan, and R. W. Liu, “MVFFNet: multi-view feature fusion network for imbalanced ship classification,” *Pattern Recognition Letters*, vol. 151, pp. 26–32, 2021.

[16] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[17] J. Wang, L. Guo, Y. Wang, L. Deng, F. Wang, and T. Li, “A vortex identification method based on extreme learning machine,” *International Journal of Aerospace Engineering*, vol. 2020, Article ID 8865001, 10 pages, 2020.

[18] Y. Lecun, L. Bottou, Y. Bengio, and Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[19] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks[J],” *Journal of Machine Learning Research*, vol. 9, pp. 249–256, 2010.