An Optimized Cosine Similarity-Based Attention Gate for Temporal Sequence Patterns Recognition

Journal of Physics: Conference Series **2622** (2023) 012012 doi:10.1088/1742-6596/2622/1/012012

An Optimized Cosine Similarity-Based Attention Gate for Temporal Sequence Patterns Recognition

Ammar Abdullah*, Nurul Ashikin Abdul-Kadir, Fauzan Khairi Che Harun

Faculty of Electrical Engineering, Universiti Teknologi Malaysia

E-mail: ammar27@graduate.utm.my

Abstract. The attention gate is often employed in various applications, such as recommender systems, to determine the correct context of the input data. Adopting the attention gate has been proven to improve prediction accuracy successfully. However, for a temporal sequence problem such as sensor-based Sign Language Recognition (SLR), it is still challenging to integrate the attention gate into the solution since the processing input is only in the form of a sine waveform. In our work, we propose an optimized cosine similarity-based attention gate to manipulate the sine wave signal while improving the recognition of temporal sequence data. We also evaluate and compare the various distance calculations and determine the best one for the SLR application. Adopting a distance-based attention gate has successfully achieved the recognition accuracy of 99% while reducing the error rate to 5%.

1. Introduction

Temporal sequence pattern recognition is a crucial field with diverse use cases. Sign language recognition (SLR) and human activity recognition are prominent implementations of this. By collecting user interaction data in time series, we can accurately predict the user's behavior or recommend something valuable to the users. There are various types of classification techniques often employed to learn the user's temporal sequence patterns, such as the Long Short-Term Memory (LSTM) [1, 2] model and Gated Recurrent Unit (GRU) [3, 4] model. However, using these solutions alone is insufficient to achieve higher prediction accuracy, especially on a large and diverse pattern of datasets. Hence, an extended layer of attention gate has been proposed to enhance the performance [4, 5]. One application is the recommendation engine for the mobile game [5], which has successfully combined different purpose attention models to build up a single hierarchical attention model. This combination can extract some useful hidden information for recommendation enhancement. Other work has proposed a co-attention model-based Recurrent Neural Network (RNN) to improve image captioning by capturing semantic concepts from the visual image [6]. This concept is similar to hierarchical attention, except they distributed the attention model into three separate components to enhance each phase.

Despite its benefits, extracting the context of temporal sequence sensory data in SLR application is challenging using the previous method. Fortunately, we can obtain some information by manipulating the distance difference between each time step in the time series. In other words, a unique pattern extracted from this can describe the context of that particular sine wave signal. Hence, we propose an optimized distance difference based on cosine similarity

to extract the information that will be consumed by the attention gate to learn a hidden context of each time-series signal. Prior to this, we investigate the different techniques of distance calculation to find the most suitable algorithm that can obtain a unique pattern. Integrating a cosine similarity-based attention gate has improved the RNN classifier for the SLR system to achieve more than 99% accuracy. The remaining section is organized as follows. Section 2 reviewed some related works and state-of-art approaches for the attention gate. Section 3 will describe our implementation, Section 4 will describe the experimental result and discussion, while Section 5 will conclude the research.

2. Related Works

2.1. Sign Language and Temporal Sequence Pattern

Sign language recognition can be divided into video [3, 7, 8], and sensor-based [4, 9] solutions. In a video-based solution, multiple images are captured in sequences for gesture recognition. Still, it suffers from background noise and is inherently difficult to be used in daily life concerning the robustness and reliability factor [4]. Sensor-based, on the other hand, can mitigate these issues well. Multiple sensors fusion deployed at the hand glove in sensor-based solutions that provides pulse reading in the form of a temporal sequence pattern. To extract the gesture information, researchers use a few methods such as Hidden Markov Model (HMM), RNN, LSTM, or GRU to learn the temporal sequence pattern from the sensors. All these implementations have been successfully demonstrated at reasonably high accuracy with a certain number of datasets, yet still not experimented against large datasets.

2.2. Attention Gates

The attention gates have been frequently used in the NLP area to assist the classifier in understanding the actual context of the sentence or paragraph. Similarly, the attention gates are adopted in SLR to learn the area of interest of the features [10]. The adoption of attention gates has been successfully tested over large datasets which consist of 500 categories. In [5], the hierarchical attention-based LSTM has been used to provide a recommendation to the users. They divided the attention hierarchy into 2; match and event attention layers. Both attention layers will extract a piece of specific information related to the respective events or matches, then accumulate to provide a suitable recommendation to the players. This map-reduce concept is suitable to handle large non-temporal datasets whereas it chunks the data into smaller portions and extracts only the specific information before accumulating to generate the final result. An encoder-decoder structure that includes the GRU and attention gate has been proposed in previous research [11]. The attention gate is used to learn the critical background of temporal sequence data and provide some influences to the hidden and output layer. In this work, a comparison has been made between attention-based GRU and other methods, whereas the proposed attention-based GRU can achieve higher prediction accuracy.

2.3. Distance Calculation

A few commonly used distance calculation algorithms have been applied in many applications, such as Euclidean distance, Cosine similarity algorithm, and Manhattan distance. Each of these algorithms uses unique metrics to obtain a distance difference, thus resulting in a diverse outcome. The cosine similarity is mainly used in the search engine, and collaborative filtering [12]. In previous work, they transformed each user's historical data into a single formatted table and computed the scalar value between each of the array of vector components by using cosine similarity. From the result, they managed to determine the closest similarity between users. Hence, the system can provide a recommendation to the user by using others' history. Euclidean and cosine-similarity are vastly adopted to compute the similarity between two temporal sequence data in vectors. Depending on the application, the former can be preferred

Figure 1. SLR System Architecture

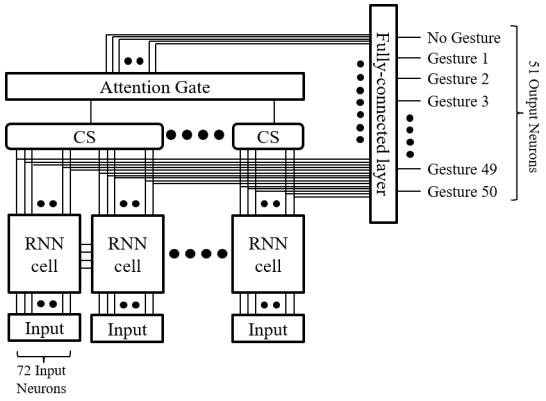

Figure 2. Proposed RNN Structure

over the latter or vice versa due to their distinct behavior. Therefore, the selection of both algorithms must be examined properly to choose the most suitable one for the system.

3. Proposed Solutions

3.1. System Architecture

In this work, we built a system for SLR applications using an IMU sensor-based approach, as depicted in Fig. 1. A set of temporal sequence data is acquired from 6 IMU sensors and pre-processed for data cleansing and sampling. The pre-processed data is then sent to the RNN classifier input to predict the gesture corresponding to the respective temporal sequence pattern. In the following subsections, we will learn further about the structure of the RNN classifier with our proposed Attention Gate mechanism, the analysis, and comparison between multiple distance calculation techniques, and finally, the backpropagation method for our proposed attention layer.

3.2. RNN Classifier with Attention Gate

The RNN classifier module comprises n-stages RNN, cosine similarity (described as CS) module, attention gate, and fully connected (FC) layer at the output stage, as depicted in Fig. 2. Each of the IMU sensors provides a 3-axes accelerometer and gyroscope readings, that sum up to 72 neurons required at the RNN input layer. The RNN hidden state comprises 100 neurons, while the attention layer comprises 256 neurons. The system is trained to recognize 50 kinds of sign language gestures. Hence, we set the output layer to have 50 gestures plus 1 for no gesture case.

The CS module is connected between two sequential RNN stages since we are interested in finding the distance between these two sequential vectors. Therefore, the number of the CS module is $n-1$, whereas n is the number of RNN stages. The CS module generates a scalar value resulting from cosine similarity calculation. The scalar outputs from $n-1$ CS modules are merged into the attention gate for processing. At the attention gate, the scalar outputs from the CS module are stored in a temporary buffer and held until it receives all outputs from CS. Then, the cosine similarity values are processed to extract hidden information from the pattern, which can influence the final prediction outcome.

Subsequently, at the final FC layer, all outputs from each RNN cell and the output from the attention gates layer are accumulated to predict the gesture from the respective temporal sequential pattern. In summary, we can define the output of the FC layer, Y_o as the addition of the attention layer, Y_{att}, and the hidden layer, Y_h.

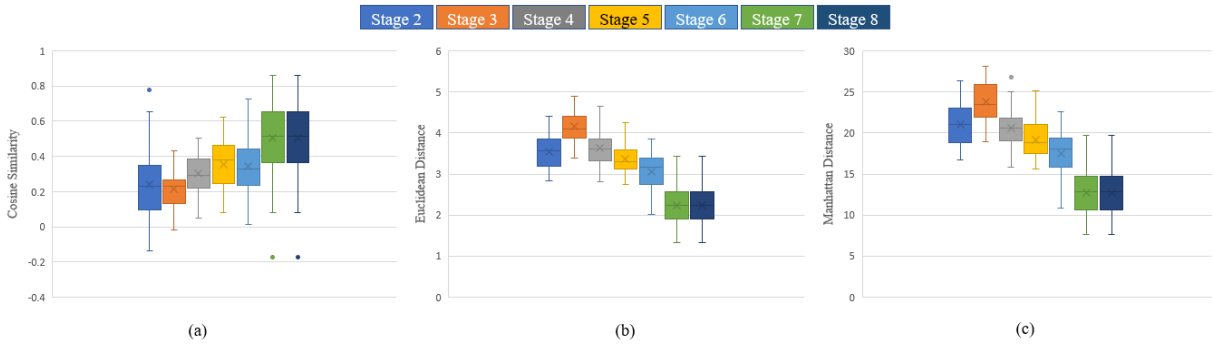

Figure 3. Errors between gestures for each method; (a) Cosine similarity, (b) Euclidean distance, and (c) Manhattan distance.

$$Y_o = Softmax(Y_{att}W_{att} + Y_hW_h + b_o) \tag{1}$$

where the W_{att} and W_h are the weight for the attention layer and hidden layer, respectively, while b_o is the output bias. The output of the attention layer can be separately defined as

$$Y_{att} = ReLU(Y_{cs}W_{cs} + b_{att}) \tag{2}$$

The CS output, Y_{cs} can be defined using the distance calculation function and described as follows,

$$Y_{cs} = D(Y_h, Y_{h-1}) \quad \text{for distinct } h \in \{2, \ldots, n\} \tag{3}$$

where the Y_h is the output of the RNN hidden state, which started from stage 2 until final stage n, and D is the distance calculation function which will be explained in the next subsection. At the hidden state, the algorithm can be defined as follows,

$$Y_h = ReLU(X_{inp}W_{inp} + b_h) \tag{4}$$

where X_{inp} is the pre-processed sensory data acquired from the accelerometer and gyroscope.

3.3. Distance Analysis

Since the time-series data has a dependency on the previous and next data, we can observe some unique patterns from the vector differences between the two timestamps. Typically the features are extracted at the input layer. However, in this work, we initially let the RNN cell learn the hidden features by itself to take advantage of the deep neural network capability. Prior to distance analysis, we let the RNN network executes training for all 50 gestures without the attention gate. Then, we extracted the data at the RNN output during the feed-forward operation for distance analysis.

From the extracted features, we compute the Euclidean distance, cosine similarity, and Manhattan distance between two timestamps from stage 2 until stage 8 and plot the error graph as depicted in Fig. 3. Each error stage's result consists of multiple executions of different gestures. In short, we can observe that, from the result, the cosine similarity algorithm shows a higher variance as opposed to other algorithms despite its scale. In other words, the cosine similarity pattern for the first to the last gesture has significant and distinct differences that the system can further learn to perform a more accurate prediction.

We further look into each algorithm to find the reason behind these findings.

3.3.1. Euclidean Distance The Euclidean distance is obtained using the following equation,

$$D(\mathbf{p}, \mathbf{q}) = \sqrt{\sum_{i=1}^{n} (\mathbf{p}_i - \mathbf{q}_i)^2} \tag{5}$$

where $\mathbf{p}$ and $\mathbf{q}$ are the two different vectors, and n is the maximum vectors element. The Euclidean value has no maximum limit, while the minimum is 0. From the equation, we can observe that each vector is squared before the addition operation. Even though the Euclidean distance is concerned about the magnitude and direction, in some cases, it seems to relax the condition. For example, if we have a set of vector $\mathbf{p}$ of $[6, 7]$ and vector $\mathbf{q}$ of $[3, 5]$ for gesture A. While for gesture B, we have a set of vector $\mathbf{p}$ of $[2, 7]$ and vector $\mathbf{q}$ of $[5, 9]$. In this case, the differences from each vector $\mathbf{p}$ and $\mathbf{q}$ is $[9, 4]$ after squared, which gains the scalar value of 3.61 for both gestures, hence losing the sense of direction.

This can be useful for application that requires only the magnitude value. However, in SLR, the gesture movement can be similar in magnitude but in the opposite direction, resulting in other types of gestures. Therefore, the less sensitive behavior to the direction has lowered the variance between some gestures, as seen in Fig. 3(b).

3.3.2. Cosine Similarity As for the cosine similarity, we calculate by using the equation as follows,

$$D(\cos(\mathbf{p}, \mathbf{q})) = \frac{\mathbf{pq}}{\|\mathbf{p}\|\|\mathbf{q}\|} = \frac{\sum_{i=1}^{n} \mathbf{p}_i \mathbf{q}_i}{\sqrt{\sum_{i=1}^{n} (\mathbf{p}_i)^2} \sqrt{\sum_{i=1}^{n} (\mathbf{q}_i)^2}} \tag{6}$$

where $\mathbf{p}$ and $\mathbf{q}$ are the two different vectors, and n is the maximum vector element, while the range is between -1 and 1. From the analysis, we denote a few elements in the cosine similarity algorithm that can produce a better result for the SLR application.

One is due to scaling and normalization behavior. As can be seen from the result, even though we require a significant variance between each vector, we want to ensure that all vectors have the same scale during the operation. To further explain, some candidates possess strong movement, thus gaining a higher input value than those who possess weak movement even when they perform the same gesture. Since we are only concerned about the pattern, the magnitudes of the same gestures have to be scaled and normalized in a batch to get the same set of patterns. Few studies have been done to evaluate the batch normalization effect on the neural network performance, which proves this concept [13].

As opposed to the Euclidean distance, the cosine similarity can handle the case as in the example above due to its multiplication behavior. The cosine similarity result shows the difference between 0.97 and 0.99 for the respective gesture A and B. This value can be significantly different in extreme cases, especially in a multidimensional case, thus able to generate a unique input pattern for the attention gate to run the processing further. The variance can be depicted in Fig. 3(a).

3.3.3. Manhattan Distance The equation for Manhattan distance can be viewed as follows,

$$D(\mathbf{p}, \mathbf{q}) = \sum_{i=1}^{n} \|\mathbf{p}_i - \mathbf{q}_i\| \tag{7}$$

where $\mathbf{p}$ and $\mathbf{q}$ are the two different vectors, and n is the maximum vector element, while the range is similar to Euclidean distance from 0 to unlimited. The characteristic of Manhattan distance is converting the operation into an absolute value. Therefore, we can assume the

behavior is similar to the Euclidean distance; in some cases, it may also neglect the direction. From Fig. 3(c), we can observe the pattern similarity between Euclidean and Manhattan distances.

3.4. Network Training Method

The network cannot be trained only in one cycle repetitively. We must ensure that clean and pre-trained data is used to train our attention model. Otherwise, it will lead to a wrong optimization at the attention layer iteratively, thus, accumulating the error. Furthermore, by running simultaneously, it has the possibility for the vanishing gradient to occur due to the deep structure of the network. Hence, not much improvement can be seen even by implementing the attention gate at the full system level. Therefore, to train our network, we break the backpropagation operation into two cycles; the main network and the attention layer cycle.

3.4.1. Backpropagation for the main network At the main network, we train the neurons at the RNN cell and part of the FC layer, while freezing the CS and attention gate layer. The training uses an Adagrad optimizer with a 1.0 learning rate and entropy loss calculation. The purpose of the main network training is to evaluate the error cost function and perform weight optimization only at the RNN cell and part of the FC layer separately. Additionally, we can obtain the pre-trained RNN auxiliary output that can be used for the next training cycle at the attention layer.

3.4.2. Backpropagation for the attention layer Each time the first cycle for main network backpropagation is completed, the system executes the second cycle training by freezing the RNN cell layer. The only affected layers during the second cycle are the CS layer, Attention Gates, and the remaining part of the FC layer. We use the same Adagrad optimizer but need to fine-tune the learning rate to find a suitable value that can prevent a vanishing gradient from occurring. In this case, the optimized learning rate is 0.5. The weight and bias optimization are done separately only on the respective neurons.

4. Experimental Result And Discussion

4.1. Data Collection

We have collected the data for 50 different Malaysian Sign Language (MSL) gestures from 20 subjects and each subject has to perform 25 gestures, which must be repeated five times.

4.2. Result And Analysis

For a performance comparison, we attempt to run training and testing for 5 cases with different conditions, as depicted in Table 1. The first one is by using the RNN without any attention gate. Then, we added the attention gate, but without any distance calculation module in between the RNN cells and the attention layer. Thus, the attention gate will feed directly and process all outputs from RNN cells. The third, fourth, and fifth conditions integrate the CS, Euclidean, and Manhattan distance modules, respectively. From the result, we can observe that without the attention gate, initially, the system can only get an accuracy of more or less around 96.8%. Merely adding the attention gate without any strategy does not help, and need to optimize the attention gate neurons number further to improve the performance.

In the case of using Euclidean and Manhattan distance, it seems the accuracy can be slightly improved since the algorithm does extract some insignificant features for the attention gate to be processed. At the same time, it does improve the loss by 0.06 - 0.08 compared to the without-attention gate case. The CS-based attention gate can significantly improve the accuracy by 98.2% and reduce the loss to 0.091. This observation is aligned with our expectation that the

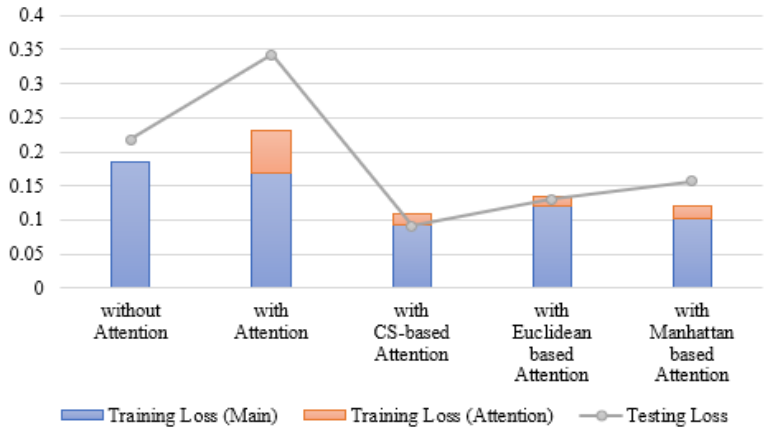

Figure 4. Training and testing loss.

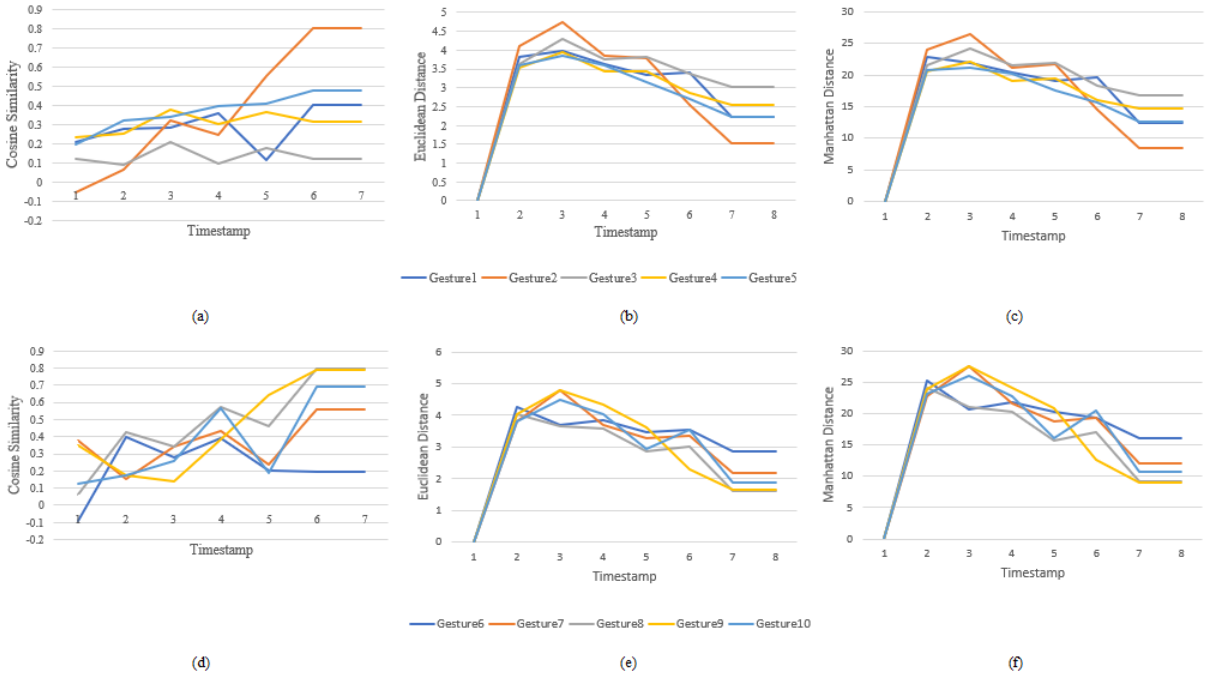

Figure 5. (a), (b) and (c) are the cosine similarity, Euclidean distance, and Manhattan distance result for gesture 1 to 5 respectively, while (d), (e) and (f) are the cosine similarity, Euclidean distance and Manhattan distance result for gesture 6 to 10 respectively

CS can extract a more unique pattern than Euclidean and Manhattan, thus providing a better result. The pattern for each algorithm is shown in Fig. 5 for selective gestures 1 to 10.

5. Conclusion

From the research, we can conclude that integrating a distance calculation technique can serve as the hidden features extractor for our attention layer in the case of temporal sequence input. To determine which technique can provide optimum output, especially to the SLR use case, we compare three well-established distance calculation techniques and observe that all can be used to optimize the system further. As a result, the cosine similarity can perform better compared

Table 1. The recognition accuracy of each implementation.

Structure	Test Accuracy (%)	Loss
RNN without Attention Gate	96.8	0.218
RNN with pure Attention Gate	93.0	0.342
RNN with CS-based Attention Gate	99.2	0.091
RNN with Euclidean-based Attention Gate	98.2	0.130
RNN with Manhattan-based Attention Gate	97.5	0.157

to others. Our results show the capability of this implementation that can achieve more than 99% while reducing the loss to 0.091.

Acknowledgments
This work was supported and funded by the Ministry of Higher Education and Universiti Teknologi Malaysia under Research University Grant No. Q.J130000.2601.14J50.

References
[1] Jerrin T P, Pradeep K G, Ramakrishnan A G, and Kanishka S 2021 Automated classification of EEG into meditation and non-meditation epochs using common spatial pattern, linear discriminant analysis, and LSTM *IEEE Region 10 Conf. (TENCON)* 215-218
[2] Xianbo Z, Chao W, Jing Z, Feng L, and Zezhou L 2022 Attention Based CNN-LSTM Network for Anomaly Pattern Classification of Multivariate Time Series *IEEE 11th Data Driven Control and Learning Syst. Conf. (DDCLS)* 1015-1020
[3] Seunghyeok S, and Whoi Y K 2020 Skeleton-Based Dynamic Hand Gesture Recognition Using a Part-Based GRU-RNN for Gesture-Based Interface *IEEE Access* **8** 50236-50243
[4] Ghazaleh K, Pyeong G J, Yacine A, and Samer M 2021 Skeleton-Based Dynamic Hand Gesture Recognition Using a Part-Based GRU-RNN for Gesture-Based Interface *IEEE Trans. on Automation Sci. and Eng.* **18** 495-507
[5] Qiongjie Y, Xiaofei L, Hai J 2019 Hierarchical attention based recurrent neural network framework for mobile MOBA Game recommender systems *Proc. - 16th IEEE Int. Symp. Parallel Distrib. Process. with Appl. 17th IEEE Int. Conf. Ubiquitous Comput. Commun. 8th IEEE Int. Conf. Big Data Cloud Comput. 11t* 338–345
[6] Bin Z, Xuelong L, and Xiaoqiang L 2019 CAM-RNN: Co-Attention Model Based RNN for Video Captioning *IEEE Trans. Image Process.* **28** 5552–5565
[7] David G L, Jade G, Sreenivasa R Y, Daniel R, Romuald M, Om J P, Linga R C 2022 Video Hand Gestures Recognition Using Depth Camera and Lightweight CNN *IEEE Sensors Journal* **22** 14610-14619
[8] Okan K, Thomas L, Yao R, Neslihan K, Gerhard R 2020 DriverMHG: A Multi-Modal Dataset for Dynamic Recognition of Driver Micro Hand Gestures and a Real-Time Recognition Framework *15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020)* 77-84
[9] Xianzhi C, Jiang L, Shigeru S 2021 A Sensor-Based Hand Gesture Recognition System for Japanese Sign Language *IEEE 3rd Global Conf. on Life Sci. and Tech. (LifeTech)* 311-312
[10] Jie H, Wengang Z, Houqiang L, and Weiping L 2021 Attention-Based 3D-CNNs for Large-Vocabulary Sign Language Recognition *IEEE Trans. Circuits Syst. Video Technol.* **29** 2822–2832
[11] Guixiang X, Jiancai S, Xiangfei K, Yu P, Chengying Q, and Han L 2019 Prediction of Natural Gas Consumption for City-Level DHS Based on Attention GRU: A Case Study for a Northern Chinese City *IEEE Access* **7** 130685–130699
[12] Xiuli W, Zhuoming X, Xiutao X, Chengwang M 2017 Computing user similarity by combining SimRank++ and cosine similarities to improve collaborative filtering *14th Web Inf. Syst. Appl. Conf. WISA 2017* 205–210
[13] Shuang W, Guoqi L, Lei D, Liu L, Dong W, Yuan X, Luping S 2019 L1 -Norm Batch Normalization for Efficient Training of Deep Neural Networks *IEEE Trans. Circuits Syst. Video Technol.* **30** 2043–2051