Lightweight Long and Short-Range Spatial-Temporal Graph Convolutional Network for Skeleton-Based Action Recognition

HONGBO CHEN\textsuperscript{1}, MENGLEI LI\textsuperscript{1}, LEI JING\textsuperscript{2}, (Member, IEEE),
AND ZIXUE CHENG\textsuperscript{2}, (Member, IEEE)

\textsuperscript{1}Graduate School of Computer Science and Engineering, The University of Aizu, Aizuwakamatsu, Fukushima 965-8580, Japan
\textsuperscript{2}School of Computer Science and Engineering, The University of Aizu, Aizuwakamatsu, Fukushima 965-8580, Japan

Corresponding author: Lei Jing (leijing@u-aizu.ac.jp)

\section*{ABSTRACT} In skeleton-based human action recognition domain, the methods based on graph convolution networks have great success recently. However, most graphical neural networks consider the skeleton as a spatiotemporally uncorrelated graph and rely on a predetermined adjacency matrix, ignoring the spatiotemporal relevance of human actions and taking up significant computational costs. Meanwhile, the methods use graph convolution to focus too much on the neighboring nodes of the joints and ignore the totality of the action. In this work, we propose a lightweight but efficient neural network called NLB-ACSE based on the Graph Convolutional Network (GCN). Our model consists of two large branches: non-local block branch that focuses on the long distance features and adaptive cross-spacetime edge branch that focuses on the short distance features. Both branches extract information across time and space, and focus on long and short information. Some simple but effective strategies also are applied to our model, such as semantics, maxpooling, and fusion inputs, which have small parameter burden but obtain a higher accuracy on ablation study. The proposed method with an order of magnitude smaller size than most previous papers is evaluated on three large datasets, NTU60, NTU120, and Northwestern-UCLA. The experimental results show that our method achieves the state-of-the-art performance.

\section*{INDEX TERMS} Skeleton-based action recognition, graph convolution network, fusion inputs, non-local block, adaptive cross-spacetime edges.

\section*{I. INTRODUCTION} In recent years, human action recognition has many applications in the real world, such as human-computer interaction, robot technology, and health care systems \cite{1}–\cite{3}. Compared to RGB videos, skeleton-based human action recognition get much attention since its robustness against complicated background and have a high computational efficiency \cite{4}, \cite{5}. For the above reasons, our work focuses on the task of skeleton-based action recognition.

Earlier methods \cite{6} simply extract feature vectors from the coordinates of joints as input, and lack of mining of original information, which also limited the accuracy of the model. With the rise of deep learning, the earlier deep-learning-based methods in this field use Recurrent Neural Networks (RNN) or Convolutional Neural Networks (CNN), researchers manually construct skeleton data as pseudo-images \cite{7} or coordinate sequences vectors \cite{8} as input to the neural network, which have achieved much better performance than previous methods. However, these methods overlook the Non-Euclidean graph-structured between the human joints, which cannot capture the inherent spatial relationship between joints.

Recently, the graph-based methods attract much attention due to their superiority to model the relations between body and joints. Yan \textit{et al.} \cite{9} propose ST-GCN to model the skeleton data with graph convolutional networks (GCNs), which contains spatial graph convolution and temporal graph convolution. The ST-GCN improves the accuracy of action recognition to a new level, and substantial ST-GCNs are subsequently proposed based on it \cite{10}, \cite{11}. However, there are still three problems to be addressed in these methods.

The first problem lies in the way of the feature extraction of the graph convolution. ST-GCN performs feature extraction by dividing the human body into five parts (left arm,
right arm, left leg, right leg and torso), however an ideal algorithm to extract features from graph convolution should look beyond local link dependencies and integrate global information, since structurally unrelated joints may also have a strong correlation in action. For example, the clapper is intuitively composed of disconnected joints that are part of different parts in ST-GCN. Some of the ways [12] to improve the ST-GCN approach to extract global features by performing graph convolution are obtained by extracting high-order polynomials of the adjacency matrix of the skeleton. That is, capturing the number of wanderings of each pair of nodes through the power adjacency matrix. Although the ability to expand the sensory field and extract information from more distant nodes to achieve graph convolution somewhat alleviates the lack of local information. However, cyclic walks on skeleton graphs mean that the closer nodes have higher weights. The bias weights make the weights of closer node cycles much larger than those of more distant nodes, which results in higher polynomial orders under performing in terms of their ability to extract features. This approach is helpful to extract global information, but there are still problems. Therefore, the aggregation of global information remains a major challenge for the field.

The second problem lies in the spatial-only and temporal-only modules (Figure 1). However, to this end, most existing approaches [10], [11] follow ST-GCN are to first use graph convolutions to extract spatial relationships at each time step, the temporal dynamics are then modeled using 1D convolutional layers. Figuratively, the current approach is more like splitting the 3D Spatial-temporal convolution into a 2D spatial graph convolution and a 1D temporal convolution. Although modeling the decomposition of the spatial and temporal domains is very capable of extracting features effectively, it also hinders the exchange of information between the spatial and temporal domains. Joints with strong spatial-temporal dependencies become progressively less relevant with this spatial-temporal separation in the propagation process. So, this structure does not capture the strong dependencies in action space-time well and may overlook powerful cues used for prediction when distinguishing similar actions.

The third problem lies in a very large number of participants. In particular, the 2s-AGCN [13] introduces a multi-stream branching structure, many recent efforts [10], [14] rely on multi-stream branching to achieve accuracy while also multiplicative growth the number of parameters. In addition, to obtain a larger perceptual field, ST-GCN uses a larger convolutional kernel (1 × 9), which further increases the memory burden. In fact, there is still a large gap between the larger number of participants and the application in real world.

In this work, we address the above limitations from three aspects. First, we use non-local block (Figure 2 (left)) to capture space-time global features. Compared with using local graph convolution, non-local block is better at capturing long-range dependencies on space-time, thus achieving a global perceptual field for each joint. Also by cross-space-time self-attention, the space-time significance is further captured as well. Second, we use adaptive inter-temporal 3D edges (Figure 2 (right)) to keep joints densely connected to their 1-hop spatial neighbors to capture short distance features in space-time. By combining the two modules in space-time we extract both long and short distance feature. The fusion of long and short information will significantly facilitate the extraction of spatiotemporal information. Third, we use multi-stream input branching, which effectively reduces the parameter size. And using the max-pooling layer in combination with semantics (frame index and joint type) increases the recognition rate without significantly increasing the parameters. In order to obtain a larger perceptual field and to minimize the parameter burden, we use different dilation windows to gather temporal information. With light-weight model size than most previous works, our model achieves the state-of-the-art performance on the NTU60 [5], NTU120 [15], and SYSU datasets [16]. The main contributions of this work are summarized as follows:

- Our proposed model extracts information across time and space, bridging the gap between spatiotemporal separation models.
- Our proposed model is able to capture both short and long-range dependencies in space-time, which will help to form a more refined sensory field.
- We propose a number of strategies to reduce the parameters, our model on a smaller number of parameters achieves the state-of-the-art performance on three large-scale datasets for skeleton-based action recognition.
II. RELATED WORK
In this section, we briefly review methods that are dedicated to the skeleton-based action recognition task in detail. Due to the lack of generality in model architecture, the hand-craft [6] methods are mostly replaced by deep learning-based methods. So we divide these deep learning-based methods into two categories: non-GCN-based and GCN-based.

A. NON-GCN-BASED METHOD
In the early days of skeleton-based human action recognition, methods based on Recurrent Neural Network (RNN) and Convolutional Neural Network (CNN) have been widely used. The RNN, represented by Long Short-Term Memory (LSTM) [17] or Gated Recurrent Unit (GRU) [18], often be used to model temporal dynamics of skeleton data [19]. Zhang et al. [20] proposed a viewpoint adaptive network based on LSTM, which can automatically select the best angle to reduce the influence of different angles. Liu et al. [21] proposed a spatial-temporal LSTM model to take advantage of the contextual relevance of joints in the temporal and spatial domain. However, due to the problem of gradient explosion and disappearance, RNN is difficult to train, so this also limits the further development of RNN-based methods. Compared with RNN, CNN is easier to train and to take parallelism. Naturally, it constructs the skeleton sequence as R, G, B channels pseudo-image [22], or using 3D CNN in the task [23], which can simultaneously extract spatial and temporal features. In order to learn local and global co-occurrence features, Li et al. [24] proposed a network that first encodes point-level information, and then aggregates global-level information through a transposition operation. Before ST-GCN [9] has put forward, the performance of this network was the best. However, both the RNNs and the CNNs ignore that human skeleton structure are non-Euclidean data, the ability of the network to extract features is limited and cannot be further improved. In addition, the excessive parameter burden further shackles their development prospects.

B. GCN-BASED METHOD
Graph convolutional network [12] is widely used in the processing of graph structured data, and it is also used to model skeletal data. As ST-GCN [9] expands the spatial map structure according to human bones, the method based on GCN has gradually attracted people’s attention. However, because the adjacency matrix is based on prior knowledge, the dependence of the non-connected joint pairs cannot be well grasped. In particular, there are global features of ST-GCN that cannot capture the nodes. Since then, some methods to improve ST-GCN have been proposed. Wen et al. [25] proposed a motif-based GCN, in which the adjacency matrix is defined according to the Euclidean distance between joint pairs to model connected and disconnected joints. Tang et al. [26] defined physically disconnected edges and connected joint pairs to better construct graphics, which made up for the shortcomings to a certain extent. SR-TSL [27] took a different approach, using a data-driven method instead of using the definition of humans to detect five human body parts in each frame. Thomas et al. [12] propose higher-order polynomials of the skeleton adjacency matrix, although the global features are extracted, the problem of having bias weights limits the performance. The two-stream GCN model [13] learns the content adaptive graph, and carries out message passing through the non-local module. However, the information semantics is not used to learn the edge of the graph and the message passing, which will reduce the efficiency of the network and cause the model to be bloated. At the same time, the addition of extra streams increases the accuracy but exponentially increases the parameter burden. Therefore, Zhang et al. [28] propose a smaller parameter model, using semantics-guided GCN, and achieved relatively good results, and which successfully open up a new perspective in the field of small parameters.

After summarizing the work of the predecessors, we find two problems. One of them is the inability to extract spatiotemporal information comprehensively, which leads to the limitation of the generalization ability of the model. Another one is that the parameters of the model are too redundant, which is not conducive to training and deployment. Different from their methods, we balance parameter size and accuracy by using spatial-temporal convolution to capture separately both short-range joints dependencies and distant joints relations.

III. METHODS
In this section, we introduce the pipeline and the components of our proposed NLB-ACSE in detail. In Figure 3, we show the overall structure of our model. First, we introduce the fusion multi-stream inputs. Next, we introduce non-local blocks and adaptive cross-spacetime edges, respectively, and finally we introduce the role of semantic collocation max-pooling layer in the model.

A. NETWORK ARCHITECTURE
Figure 3 illustrates the architecture of NLB-ACSE. First, we fuse the three flows of joint flow, velocity flow, and bone flow. We splice the semantics into the channel so that the dimension of the output feature map rises to 128, and then input the non-local blocks and ACSE blocks respectively. Finally, we sum up the results through a 3 × 1 convolutional layer with four different dilation windows and a maxpooling layer to output the final result.

B. FUSION MULTI-STREAM INPUTS
Inspired by the success of SGN [28], multi-stream fusion on the inputs side has lighter parameters than inheriting the multi-stream-based methods [10] using multiple inputs (equivalent to multiple network branches). By using pre-branching fusion, as much hidden information as possible is effectively mined before the input network to help capture the action, and more importantly, the parameter burden is
compressed to a very small size. Compared with the simple use of position and velocity by SGN, we added bone features flow and used more details. In this work, the input features are divided into three flows according to fusion methods: 1) joint flow 2) velocity flow and 3) bone flow.

In this work, the original 3D coordinate set of an action sequence \( \{C \times T \times V\} \), where C, T, V denote the coordinate (C equals to 3 due to 3D), frame and joint, respectively. So naturally we can get information about the position of the whole skeleton, denote as \( P_{T \times V} \). Similarly, we can use the coordinates of the center of gravity \( V_i \) to get the relative position \( P_{T \times V} = P_{T \times V} - P_{T \times V_{i}} \). We splice it on the channel to get the joint flow \( P \). At this time, the number of channels is 6.

Similarly, it is simple to get the velocity information from the position information. We define the velocity information in terms of the difference on the time frame \( V_{T \times V} = P_{T \times V} - P_{T-1 \times V} \). Again, we can get the acceleration information from the velocity information, expressed as \( A_{T \times V} = V_{T \times V} - V_{T-1 \times V} \). It is worth noting that we fill the tensor after subtraction with zero and then we splice it on the channel to get the velocity flow \( V \).

Similarly, we divide the bone flow into two parts: bone length and angle. Similar to the relative position, we can define the skeleton length of a prior knowledge of the skeleton with respect to the center of gravity \( V_i \), denoted as \( B_j = P_{T \times V} - P_{T \times V} \). We define the angle by the length, denoted as \( B_a = \arccos(B_i / \sqrt{B_{i,x}^2 + B_{i,y}^2 + B_{i,z}^2} ) \), where \( x, y, z \) are expressed as three-dimensional coordinates in space. We combine the length and angle together to represent the bone flow \( B \).

Take the embedding of position, we encode the position using two fully connected (FC) layers as:

\[
\tilde{P} = \sigma(W_2(\sigma(W_1 P + b_1) + b_2))
\]  

(1)

\( W_1 \) and \( W_2 \) are weight matrices, \( b_1 \) and \( b_2 \) are the bias vectors, \( \sigma \) denotes the ReLU function. Then we add the three flows of data as input as:

\[
Input = (\tilde{P} + \tilde{V} + \tilde{B})
\]  

(2)

C. SEMANTICS

Inspired by SGN, we find that the semantics (joint type and frame index) can indeed effectively help the network to recognize actions. Briefly, the meaning of joint types (e.g. head and foot) is significant for the different actions recognized by humans, and the order of each joint’s movement in time frames is also clearly helpful for action recognition.

For the \( k_{th} \) joint \( V_k \), we represent it by a one-hot vector, where \( k_{th} \) dimension is one and the others are all zeros. Similarly, for the \( k_{th} \) frame \( T_k \), we also represent it by a one-hot vector, where \( k_{th} \) dimension is one and the others are all zeros. In simple terms, the 1’s on the matrix correspond to different joints and frames, and the other positions are filled with 0’s. Just like in natural language processing (NLP) where words are encoded, we give them semantic meaning so that they understand the action as if it were a sentence. A graph of the semantics is given in Figure 4. Then we can easily obtain the joint semantics with output feature map V and the frame index with output feature map T. We make it pass through the fully connected layer of Equation 1, output feature map 64, and sum the two parts.

\[
\tilde{S}_j = \sigma(W_2(\sigma(W_1 S_j + b_1) + b_2))
\]  

(3)

\[
\tilde{S}_f = \sigma(W_2(\sigma(W_1 S_f + b_1) + b_2))
\]  

(4)

\[
\tilde{S} = \tilde{S}_j + \tilde{S}_f
\]  

(5)

\( W_1 \) and \( W_2 \) are weight matrices, \( b_1 \) and \( b_2 \) are the bias vectors, \( \sigma \) denotes the ReLU function. \( \tilde{S}_j \) and \( \tilde{S}_f \) denote the joint type and frame index, \( \tilde{S} \) is the final output. Note that the semantics and fusion inputs are coded into the same feature space, so that we can concat it all together.

D. NON-LOCAL BLOCK

A human skeleton graph is denoted as \( G = (V, E) \), where \( V = \{V_1, V_2, \ldots, V_n\} \) is the set of N nodes representing joints, and E is the edge set representing bones captured by an adjacency matrix A. According to [9], the spatial
GCN operation for each frame $t$ in a skeleton sequence is formulated as

$$F_{out} = \sum_{d=0}^{D} W_d F_{in}(\tilde{D}_d^{-\frac{1}{2}} \tilde{A}_d \tilde{D}_d^{-\frac{1}{2}} \otimes M_d)$$

where $D$ is a predefined maximum graph distance, $F_{in}$ and $F_{out}$ denote the input and output feature maps, $\otimes$ means element-wise multiplication, $\tilde{A}_d$ denotes the $d$-th order adjacency matrix that marks the pairs of joints under a graph distance $d$, $\tilde{D}$ is the normalize matrix of $\tilde{A}$, $M_d$ is a learnable parameter, which are utilized to tune the importance of each edge. The term $F_{in} \tilde{D}_d^{-\frac{1}{2}} \tilde{A}_d \tilde{D}_d^{-\frac{1}{2}}$ can be intuitively interpreted as an approximate spatial mean feature aggregation from the direct neighborhood followed. In the human skeleton graph, neighbors are joints adjacent to the nodes. Intuitively, it is difficult to extract motion recognition of the left hand to the right hand, which is not connected on the skeleton map, and to extract information about the foot, which is spatially distant.

So we propose to use non-local block instead of the traditional graph convolution module to extract the global information of joints on space-time. Inspired by Wang et al. [29] consider the weighting of all positions in space-time on RGB videos, we put the edge weight from the $i$th joint to the other joints (replace the others with one of the joints $j$) in the same frame $t$ modeled by their similarity in the embedded space as:

$$A(x_i, x_j) = \theta(x_i)^T \phi(x_j)$$

where $\theta$ and $\phi$ denote two transformation functions, each implemented by a $1 \times 1$ fully connected (FC) layer. $\theta(x_i) = W_3 x_i + b_3$ and $\phi(x_i) = W_4 x_i + b_4$. We obtain the new feature matrix $A(x_i, x_j)$ by mapping all joints to the feature space. Compared with the original adjacency matrix, the new matrix takes into account the influence interactions of all joints, thus weakening the disadvantage of too much weight of the proximity joints. For the skeleton sequence $\{C \times T \times V\}$, after passing through the respective fully connected layers, we will rescale the tensor so that $T, V$ concatenate the new dimensions after dot product. We apply self-attention in a new dimension, denoted as:

$$F = \text{Softmax} \ast A(x_i, x_j)$$

where Softmax denote the softmax function to act as a self-attention. Screening by self-attention helps networks more easily detect differences in actions, then we use the residual connection to form our final non-local block, denoted as:

$$\tilde{F} = G \ast F + X$$

where $\tilde{F}$ denotes the output feature maps, $G$ is transformation matrix and $X$ is residual term.

### E. ADAPTIVE CROSS-SPACE-TIME EDGES

Most existing works treat skeleton actions as a sequence of disjoint graphs where features are extracted through spatial-only (e.g. GCNs) and temporal-only (e.g. TCNs) modules. We argue that this spatial-temporal separation approach to extracting features is not as effective. It is obvious that the strong spatial-temporal relationship between two joints in the recognition action will weaken the connection with the extraction method of spatial-temporal separation, which does not facilitate the extraction of effective features. Therefore, we propose the adaptive cross-space edges as Figure 2(right) to extract the information of short-range on space-time.

We first consider how to apply local graph convolution to space instead of extracting features with spatio-temporal separation. Inspired by shift-gcn [10], we propose that there could be an adaptive sliding window that would slide through spacetime like a naive shift operator to extract spatio-temporal information. With a shift up and down operator of 1, we fill the empty part with zeros to get a sliding window of 3 as illustrated in Figure 5.

We perform two shifts of 1 to the feature map and fill the empty positions, which are the white positions in the map, with zeros to obtain a time window of length 3. Similarly, if the shifts are of length 2, then a time window of length 5 can be obtained, and so on.

So we consider a sliding temporal window of size $\alpha$ over the input graph sequence, which at each step we can easy obtain a spatial-temporal sub-graph $G_{\alpha} = (V_{\alpha}, E_{\alpha})$, where $V_{\alpha} = V_1 \cup V_2 \ldots \cup V_{\alpha}$ is the union of all node sets across $\alpha$ frames in the window. In simple terms, the channel is decomposed into the original $\alpha$ copies, and each copy is slid in time and space, with each node forming a new edge with
its neighbors in space as shown in Figure 2 (right). The initial edge set \( E_\alpha \) is defined by tiling \( \tilde{A} \) into a block adjacency matrix \( \tilde{A}_\alpha \), where \( \tilde{A}_\alpha = \alpha N \times \alpha N \) by sliding each window. Then we can easily obtain the new Equation 6, denoted as:

\[
F_{\text{out}} = \sum_{d=0}^{D} W_d F_{\text{in}} (\tilde{D}_\alpha^{-\frac{1}{2}} \tilde{A}_\alpha \tilde{D}_\alpha^{-\frac{1}{2}} \otimes M_d) \tag{10}
\]

In general, it is easy to define the value of \( \alpha \) for integers. However, the disadvantage of doing so is that the feature map with a single sense field and not necessarily suitable for all dimensions. So we scale to real numbers and use linear interpolation to let the network learn adaptively the size of \( \alpha \) denoted as:

\[
\tilde{F}_{\text{out}} = \lambda F_{[T+\alpha]} + (1-\lambda) F_{[T+\alpha]+1} \tag{11}
\]

\( \lambda = \alpha - \lfloor \alpha \rfloor \) denote as the remainder resulting from realizing the integers. Since the anchor point falls between \( \lfloor T+\alpha \rfloor \) and \( \lfloor T+\alpha \rfloor + 1 \) after realization, interpolation is performed between this interval. In contrast to fixed values, linear interpolation of values provides an adaptive range of real numbers, while learnability ensures that the network can adaptively choose the optimal solution. We have validated the effectiveness of our improvements through ablation study.

**F. TCN MODULE WITH DILATION**

In order to obtain a larger perceptual field while reducing the number of parameters, we use a \( 3 \times 1 \) fixed convolution kernel and divide the channel domain into four parts. The convolution kernels with different scales of dilation as 0, 1, 2, and 3 are split as shown in the Figure 6.

Using convolution with dilation gives a larger perceptual field with the same parameters as regular convolution.

**G. MAXPOOLING LAYER**

The maxpooling layer plays a very important role in our model. Different from most models use mean pooling, the joint semantics of the maxpooling layer can play a role similar to that of the attention module and, more importantly, impose a largely negligible parameter burden. We visualise the throwing action as shown in Figure 7.

We can visualise the throwing motion through the maxpooling layer and output circles of different sizes according to the size of the joint influence. It can be clearly seen that the throwing arm, the head and the torso play a huge role at this moment.

**IV. EXPERIMENTS**

In this section, we evaluate the performance of the proposed NLB-ACSE on three large-scale datasets NTU RGB + D 60 [5], NTU RGB + D 120 [15] and Northwestern-UCLA [16]. Ablation studies are also performed to validate the contributions of each component in our model. Then, we compare our model with the other state-of-the-art approaches on three datasets.

**A. DATASETS AND EXPERIMENT SETTINGS**

1) NTU60

**NTU60 RGB + D Dataset [5]:** This dataset is captured by 3 Microsoft Kinect v2 cameras at the same time. The number of joint points collected by this dataset is 25, and there are 17 combinations of camera placement positions, consisting of 56,880 action clips, including 40 actors performing 60 action categories. The author of the data set suggests to evaluate the model in the following two ways: (1) Cross View(CV), where the samples of cameras 2 and 3 are used for training, and the samples of camera 1 are used for testing. (2) Cross Subject(CS), where half of the subjects in the sample are used for training, and the remaining samples are used for testing.
2) NTU120

**NTU120 RGB + D Dataset [15]:** This dataset is the original dataset expansion, the camera placement combination has 17 expansions to 32, the action classification was expanded from the original 60 behaviors fill up to 120 categories, the number of actors has been expanded to 106, the segment was expanded to 114480, and the number of joints remained unchanged. The author of the dataset suggests evaluating the model in the following two ways: The author of the data set recommends reporting model performance in two of the following situations Setting: (1) Cross Setup (C-Setup), the samples with even ID from the camera setting are used for training, and the rest are used for testing. (2) Cross Subject (C-Sub), 106 subjects are used for training and the remaining samples are used for testing.

3) NORTHWESTERN-UCLA

**Northwestern-UCLA Dataset [16]:** Is also captured by three Kinect cameras. It contains 1494 video clips covering 10 categories. Each action is performed by 10 actors. We adopt the same evaluation protocol in [32]: We use the samples from the first two cameras as training data and the samples from the third camera as test data.

4) EXPERIMENT SETTINGS

For NTU RGB + D and NTU-120 RGB + D, we use Adam with an initial learning rate of 0.001 to train the model for 120 epochs, batch size 64, the learning rate decays by a factor of 10 at the 60th epoch, the 90th epoch, and the 110th epoch, respectively. For Northwestern-UCLA, the batch size is 16, we use SGD with momentum (0.9) to train the model for 120 epochs. The learning rate is set to 0.001 and divided by 10 at epoch 60, 80, and 100. All experiments are conducted on the Pytorch platform with a GTX3060 GPU card. The data processing is similar to SGN [28].

**B. COMPARISONS WITH STATE-OF-THE-ART METHODS**

To verify the superiority and generality of our approach, in this section we compare our model with state-of-the-art methods on three datasets: NTU RGB + D dataset [5], Northwestern-UCLA dataset [16], NTU-120 RGB + D dataset [15], as shown in the following table:

| Method         | Year | X-sub(%) | X-set(%) | Params |
|----------------|------|----------|----------|--------|
| ST-GCN [9]     | 2018 | 81.5     | 88.3     | 3.10   |
| SR-TSL [27]    | 2018 | 84.8     | 92.4     | 19.07  |
| AS-GCN [30]    | 2019 | 86.8     | 94.2     | 6.99   |
| RA-GCN [31]    | 2019 | 88.7     | 94.3     | 6.21   |
| AGC-LSTM [32]  | 2019 | 89.2     | 95.0     | 22.89  |
| 4s-DGNN [33]   | 2020 | 89.9     | 96.1     | 26.24  |
| NAS-GCN [34]   | 2020 | 89.4     | 95.7     | 6.57   |
| 4s-Shift-GCN [5]| 2020 | 90.7     | 96.5     | 2.76   |
| NLB-ACSE(ours) | -    | **91.0** | **96.1** | **1.21** |

Many state-of-the-art methods utilize multi-stream fusion strategies. The first stream uses the original skeleton coordinates as input, the second stream uses the differential of spatial coordinates as input, the third and fourth streams use the differential on temporal dimension as input. And the softmax scores of multiple streams are used to obtain the fused score, such as 2S-AGCN uses former two streams of data and 4s-Shift-GCN uses the all four streams of data. It is clear that using multi-stream branch fusion has a greater parameter burden compared to our model.

On NTU60, our model performs better on a lightweight benchmark. 4s-Shift-GCN is similar to the accuracy of our model, but our parameters are only 43.8% of it.

On NTU120 and Northwestern-UCLA, NLB-ACSE obviously exceeds all state-of-the-art methods with minimum parameter size.

**C. ABLATION STUDY**

In this section, we use a model to evaluate the performance of our strategy in three aspects. First of the strategy is the performance in using semantics and pooling layers. And then we verify the effectiveness of fusion multi-stream inputs as well as adaptive spatial-temporal edges. Finally, the performance of the model is verified using single-stream branching and different number of layers respectively. The action recognition accuracies by NLB-ACSE are reported in Table 1, 2, 3, 4, respectively.

First of all, we justify our proposed semantics and maxpooling are necessary in Table 1. It is obvious to see that there is a decrease in accuracy when there is no semantics and no maxpooling, proving that both are indispensable. Meanwhile, we conduct a comparison experiment using avgpooling and find that the effect of using avgpooling alone is comparable.
to that of maxpooling, but the compatibility between mean pooling and semantics is poor, even inferior to that of using it alone.

Second, we verify the importance of each branch in a multi-stream branch in Table 2. It is clear that using three-stream fusion input is better than two-stream and single-stream with minimal parameter burden.

Third, we verify the effect of different numbers of basic blocks on the accuracy of the model in Table 3. In a smaller scale than the baseline, we verified the importance of each of the two basic blocks. At scales larger than the baseline, more basic blocks instead cause a decrease in the parameters.

Finally, we compare the gap between the adaptive $\alpha$ and the different preset values. Compared to the preset, the adaptive one has better shape-ability.

**V. CONCLUSION**

In this work, we proposed a minor parameters yet effective end-to-end neural network for high performance skeleton-based human recognition. We propose practical ways to reduce the use of parameters, and a smaller number of parameters helps to advance the field in mobile devices and implementation. We also highlight the importance of neglected spatio-temporal joint features and the need for fusion of long and short-term features in motion capture. Using experiments on three large-scale datasets, we show that our model gives quite good performance under small parameter burden. In practical applications, our model with an order of magnitude smaller model size, can be applied to security surveillance systems, health care systems, and human-computer interaction systems, etc.

**REFERENCES**

[1] Y. Zhang, C. Cao, J. Cheng, and H. Lu, “Egogesture: A new dataset and benchmark for egocentric hand gesture recognition,” IEEE Trans. Multimedia, vol. 20, no. 5, pp. 1038–1050, May 2018.

[2] L. Y. Gui, K. Zhang, Y. X. Wang, X. Li, J. M. F. Moura, and M. Veloso, “Teaching robots to predict human motion,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Madrid, Spain, 1–5, Oct. 2018, pp. 562–567.

[3] Z. Duric, W. D. Gray, R. Heishman, F. Li, A. Rosenfeld, M. J. Schoelles, C. Schunn, and H. Wechsler, “Integrating perceptual and cognitive modeling for adaptive and intelligent human-computer interaction,” Proc. IEEE, vol. 90, no. 7, pp. 1272–1289, Jul. 2002.

[4] J. Wang, Z. Liu, Y. Wu, and J. Yuan, “Mining actionlet ensemble for action recognition with depth cameras,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Providence, RI, USA, Jun. 2012, pp. 1290–1297.

[5] A. Shahroudy, J. Liu, T.-T. Ng, and G. Wang, “NTU RGB+D: A large scale dataset for 3D human activity analysis,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Seattle, WA, USA, Jun. 2016, pp. 1010–1019.

[6] R. Vemulaapalli, F. Arate, and R. Chellappa, “Human action recognition by representing 3D skeletons as points in a line group,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2014, pp. 588–595.

[7] Y. Du, W. Wang, and L. Wang, “Hierarchical recurrent neural network for skeleton based action recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Boston, MA, USA, Jun. 2015, pp. 1110–1118.

[8] C. Caetano, J. Sena, F. Bremond, J. A. Dos Santos, and W. R. Schwartz, “SkeleMotion: A new representation of skeleton joint sequences based on motion information for 3D action recognition,” in Proc. 16th IEEE Int. Conf. Adv. Video Signal Based Survell. (AVSS), Taipei, Taiwan, Sep. 2019, pp. 1–8.

[9] S. Yan, Y. Xiong, and D. Lin, “Spatial temporal graph convolutional networks for skeleton-based action recognition,” in Proc. 32nd AAAI Conf. Artif. Intell., New Orleans, LA, USA, Feb. 2018, pp. 7444–7452.

[10] K. Cheng, Y. Zhang, X. He, W. Chen, J. Cheng, and H. Lu, “Skeleton-based action recognition with shift graph convolutional network,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2nd ed. New York, NY, USA: McGraw-Hill, Jun. 1964, pp. 15–64.

[11] Y.-F. Song, Z. Zhang, C. Shan, and L. Wang, “Stronger, faster and more explainable: A graph convolutional baseline for skeleton-based action recognition,” in Proc. 28th ACM Int. Conf. Multimedia, Oct. 2020, pp. 1625–1633.

[12] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” 2016, arXiv:1609.02907.

[13] L. Shi, Y. Zhang, J. Cheng, and H. Lu, “Two-stream adaptive graph convolutional networks for skeleton-based action recognition,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 12026–12035.
