IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received September 26, 2021, accepted October 7, 2021, date of publication October 11, 2021, date of current version October 19, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3119455

Can Skeletal Joint Positional Ordering Influence Action Recognition on Spectrally Graded CNNs: A Perspective on Achieving Joint Order Independent Learning

M. TEJA KIRAN KUMAR[1], (Student Member, IEEE), P. V. V. KISHORE[1], (Senior Member, IEEE), B. T. P. MADHAV[1], (Senior Member, IEEE), D. ANIL KUMAR[2], (Member, IEEE), N. SASI KALA[3], (Member, IEEE), K. PRAVEEN KUMAR RAO[3], (Member, IEEE), AND B. PRASAD[4]

[1]Biomechanics and Vision Computing Research Center, Department of Electronics and Communication Engineering, Koneru Lakshmaiah Education Foundation (Deemed to be University), Guntur 522502, India
[2]Department of Electronics and Communication Engineering, PACE Institute of Technology and Sciences, Ongole 523272, India
[3]Department of Electronics and Communication Engineering, Kamala Institute of Technology and Science, Warangal 506009, India
[4]Department of Information Technology, Vignan's Institute of Information Technology, Visakhapatnam 530049, India

Corresponding author: P. V. V. Kishore (pvvkishore@kluniversity.in)

This work was supported by the research project scheme titled "Visual–Verbal Machine Interpreter Fostering Hearing Impaired and Elderly" through the "Technology Interventions for Disabled and Elderly" Program, SEED Division, Department of Science and Technology, Government of India, Ministry of Science and Technology, under Grant SEED/TIDE/013/2014(G).

ABSTRACT 3D skeletal based action recognition is being practiced with features extracted from joint positional sequence modeling on deep learning frameworks. However, the spatial ordering of skeletal joints during the entire action recognition lifecycle is found to be fixed across datasets and frameworks. Intuition inspired us to investigate through experimentation, the influence of multiple random skeletal joint ordered features on the performance of deep learning systems. Therefore, the argument: can joint order independent learning for skeletal action recognition practicable? If practicable, the goal is to discover how many different types of randomly ordered joint feature representations are sufficient for training deep networks. Implicitly, we further investigated on multiple features and deep networks that recorded highest performance on jumbled joints. This work proposes a novel idea of learning skeletal joint volumetric features on a spectrally graded CNN to achieve joint order independence. Intuitively, we propose 4 joint features called as quad joint volumetric features (QJVF), which are found to offer better spatio temporal relationships between time series joint data when compared to existing features. Consequently, we propose a Spectrally graded Convolutional Neural Network (SgCNN) to characterize spatially divergent features extracted from jumbled skeletal joints. Finally, evaluation of the proposed hypothesis has been experimented on our 3D skeletal action KLHA3D102, KLYOGA3D datasets along with benchmarks, HDM05, CMU and NTU RGB D. The results demonstrated that the joint order independent feature learning is achievable on CNNs trained on quantified spatio temporal feature maps extracted from randomly shuffled skeletal joints from action sequences.

INDEX TERMS Human action recognition, 3D motion capture, spectrally Graded CNNs, skeletal joint ordering.

I. INTRODUCTION

The Skeletal based action recognition is being practiced through deep learning on features extracted from 3D joint sequences. These sequences represent joint positions across a 3D action video. However, the quality of these sequences depends entirely on the capturing technologies. Two most widely used 3D human action skeleton recording systems are Microsoft Kinect and motion capture. Kinect is commercially affordable with a moderate reliability in capturing human

The associate editor coordinating the review of this manuscript and approving it for publication was Jon Atli Benediktsson.

skeleton representation as joints. On the other hand, motion capture technology is costly and capable of generating highly accurate representations of skeletal joints in 3D space.

The objective of skeletal human action recognition algorithms is to learn these 3D joint sequences and identify unique patterns for classification. Initially, joint positions were applied for training the classifiers [1], [2]. One such classifier was the graph matching (GM) algorithm [3], [4]. In GM, graph is constructed using the joint positions as nodes and the inter joint relationships as edges. Each skeletal action video frame is represented as a graph during training. Testing GM involves a computationally intensive frame by frame matching either through a learning algorithm or a matching measurement model. Similarly, decision trees [5], [6] also produced good action estimates from raw positional joint data for human action recognition on both Kinect and mocap captures.

In an ever expanding endurance for betterment in recognition accuracies, researchers saw an opportunity to develop sequence models for characterizing time series 3D joint data. Sequence modeling designs were exclusively applied to learn these 3D joint time series variations in actions for recognition. Recurrent Neural Networks(RNNs) [7] and its upgrades such as Gated Recurrent Units (GRU) [8] and Long Short Term Memory(LSTM) [9] has shown exclusive learning capabilities on sequence data. However, these networks are too deep and often need intensive computing power for execution. Hence, a successful alternative is to describe the time series joint data as a spatio temporal feature illustration [10]. In the last four years a dozen varieties of spatio temporal features have been reported on skeletal joint 3D data. These are popularly called joint feature maps characterize a human action sequences into images. Eventually, spatial patterns are learned from these action feature maps using convolutional neural networks (CNNs) for action recognition [11].

Surprisingly, most of the benchmarks works in the area of action recognition have selected different joint ordering on the skeleton during the classifier development. Interestingly, these joint orders play a key factor in determining the recognition accuracies on various datasets. To exemplify, HDM05 [12] and CMU [13], two most prominent action datasets have different joint ordering. Similarly, NTU RGB D [14] and MSRAction3D [15] are showing differences in joint ordering. This observation has profoundly influenced our research in this work. Similarly, our datasets KLHA3D102 [16], KLYOGA3D [17] and KLSLR3D [18] which are recorded using 3D motion capture (mocap) technology also show different joint orderings. So far, only a few researchers have pointed towards the impact of jumble joints on the performance of skeletal action recognition methods [19], [20]. Fig 1 shows the joint ordering across action datasets.

The idea behind this skeletal joint random order training on the deep learning networks is to learn different possible random feature representations for a robust action recognition framework. The point made by is valid as it says that the skeletal action recognition is based on joint order combination. If this order is altered by the system or forgotten by the user during data preparation, the pre-trained models are destined to give ambiguous results. In general, researchers have experienced problem during capturing 3D data using motion capture system. During capturing, the technician from multiple departments using it for their applications such as sign language, yoga poses, human action and medical biomechanics e.t.c and have given different joint orders according to their need. As they were building huge datasets for their applications over a period, different researchers were involved and resulted in different joint ordering in the skeletal datasets. When we want to test these datasets with our deep learning models, it has been found to give highly discriminate feature for within class labels. It took a while to understand the problem. Similarly, when we the models trained on 3D motion capture skeletal data and used on test inputs from Kinect data with similar number of joints has again resulted in a failed model. To convert these data pre-processing anomalies into refined information, there are two methods. One is to rerecord or reconstruct the data from scratch and the other is to use it as an opportunity to solve this problem through automation. This paper describes the research and experimentations performed for generating a research on order independent framework for action recognition.

A question that naturally arises from the above discussion is, can we design a deep network that will detect patterns in jumbled join features. Consequently this is the first work to explore the possibilities of developing a joint order independent feature learning through deep networks. Besides deep networks, features play a crucial part in overall training and improving the performance of skeletal action recognition tasks. Over the years, a variety of features [21] were computed from raw skeletal joint positions for 3D action representation. Some of these features are joint distances, angles, lines, planes, angular displacements, quadrilaterals etc.

The irony is, we actually have to maintain a constant skeletal joint order during the entire experimentation. Fig.1(a-d) shows joint ordering used in publically available benchmark 3D skeletal action datasets. Subsequently, fig's.1(e - f) are from our own yoga and action datasets. An inspection of the action skeletons in fig's.1(a -f) reveals that the joint orders were indeed different across datasets. This presents a bottleneck during comparison of a proposed deep network on these multiple action datasets. The most common form of feature representation is through the use of joint positions which intuitively are defined with respect to the skeletal joint ordering. Hence a change in joint ordering or sequencing during the recognition process effects the model accuracies as shown fig.2.

As an example, we extracted joint distance features from our previous work [22] and color coded them into RGB images using JET color coding. Fig 2 are the Joint Distance Maps (JDM) on benchmark action datasets shown in fig 1 for walking action. Next, the second row of fig 1 shows the JDM

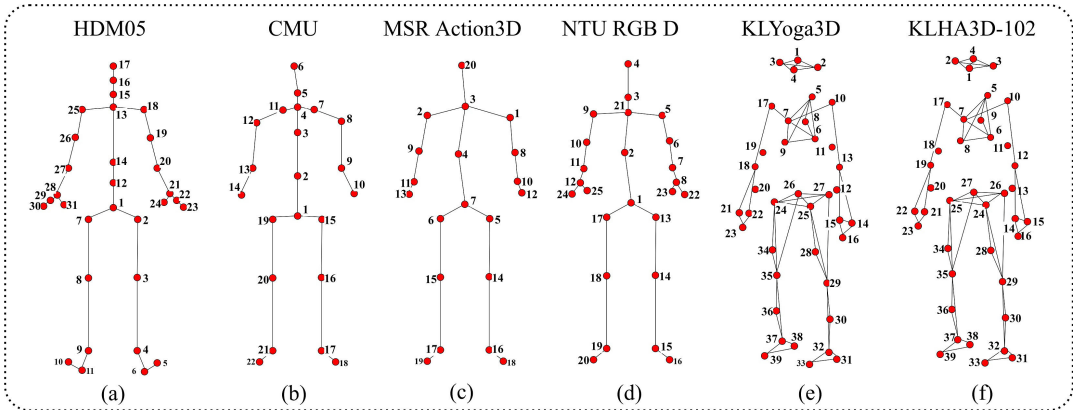

FIGURE 1. Joint ordering across action datasets.

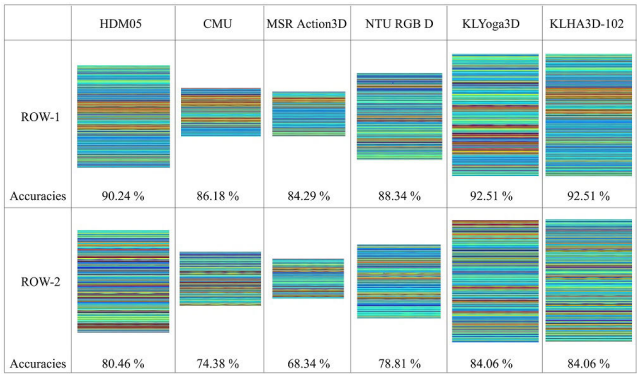

FIGURE 2. Computed recognition accuracies of JDMs under the headers same joint order testing and random joint order testing on all skeletal action datasets.

maps that are constructed with a different joint ordering than the original version in fig.1. We trained a deep CNN model from our previous work [17] on JDMs from first row of fig 2. Consequently, we tested with JDMs with same ordering and different joint ordering in second row of fig 2. Further, fig 2 shows the computed recognition accuracies under the headers same joint order testing and random joint order testing on all skeletal action datasets with a train test ratio of 15:4. In summary, the recognition accuracies were found to be below normal in all the cases when joint ordering differed in training and testing.

In this paper, we propose to develop a universal joint order independent learning network called Spectrally Graded CNN (SgCNN). Additionally, we also extend on the present feature maps into a more efficient and reliable skeletal representations. These proposed maps are called quad joint volumetric features(QJVF). The objectives of this work would be

1) To design QJV features along with a novel deep CNN model to develop a joint order independent feature learning.
2) To identify the number of randomly ordered joint feature maps required for training the designed SgCNN that results in sovereign 3D skeletal action recognition systems.
3) To determine the desirable joint feature maps that can achieve joint order independence on deep learning frameworks for 3D skeletal action recognition tasks.

The results of this study are important for attaining explicit understanding of joint ordering in 3D skeletal based action recognition on deep learning networks. The following outcomes can be expected from our experimental study on random joint order selection for skeletal action recognition:

1) A 4 joint feature map QJVF, that has shown capabilities to represent joint relationships exclusively in 3D skeletal actions when compared to existing features.
2) A refined learning network (SgCNN), with multi-dimensional filter rotations generalize the input by preventing feature loss in the dense layers..
3) A discovery on a potentially optimal feature subset that can achieve joint order independent learning on deep networks.

The rest of the manuscript is organized as follows. The following section describes various features and methods that were developed previously for skeletal based action recognition. The third section illustrates the methods developed in the work. The penultimate section presents results, discussion and analysis of various experiments conducted to achieve the formulated objectives. Finally, section V concludes the proposed problem with obtained results.

II. BACKGROUND

Human skeletal data is more robust than other modalities such as RGB video and depth. The robustness to 3D skeletal action data is because of its independence towards video backgrounds and human subject inconsistencies. These characteristics have made the 3D skeletal representation of human actions and activity, the preferred input modality for classification problems. This trend is fuelled by the availability of inexpensive hardware sensors such as Microsoft Kinect and Intel real sense 3D capture system [23], [24]. On the other hand more expensive and accurate capture technology is a multi-camera 3D mocap system [25]. 3D human

action data from these systems has revolutionized the human action recognition in the last five years. Although multitude of action recognition algorithms were proposed on these datasets [13], [26], [27], we prefer to review works focused on deep learning frameworks only.

The perfect recipe for 3D skeletal based action recognition is a combination of joint action data and the deep learning. Compared to other action data modalities it is observed that the skeletal data is spatially relational, temporally compatible and also form spatio temporal structures. Eventually, the machine learning algorithms have shown evidences to learn one or more of these characteristics for automated skeletal action recognition [28]. The first machine learning models focused on learning temporal patterns by extracting joint variations across frames [29] which further evolved by characterizing them as time series representations [10]. However the above models learned these temporal variations specific to a dataset and could not transfer the gained knowledge during testing with a different dataset. The recurrent ML models were found to be on the downshift across datasets. Hence to develop actionable Intelligence across datasets, deep learning architectures were applied on skeletal action data. Erstwhile deep learning models on vision computing applications [30] has shown impressive performances in decoding spatial and spatio temporal patterns.

Deep learning methods such as Recurrent Neural Networks(RNNs) [31], Long Short Term Memory(LSTM) [32], Convolutional Neural Networks (CNN) [33], Recurrent CNN (RCNN) [34] and lately the Graph Convolutional Networks (GCN) [35] has shown a monumental growth in human action recognition with skeletal datasets [12]–[15]. Primarily, the naturally occurring skeletal joint temporal cues in human actions are exceptionally well characterized by RNNs [31]. The structure of RNNs allow them to identify joint patterns by generating relationships between the previous and present joint variations across action sequences. Despite successful performances on skeletal action datasets, RNNs showed limitations in processing long sequences due to vanishing gradients problem [31]. This drawback was succeeded by inducing memory cells into the current architecture of RNNs to create upgrades such as Long Short Term Memory (LSTM) and Gated Recurrent Units (GRU).

LSTMs were most exclusively applied for skeletal action recognition tasks in unidirectional [32] and bidirectional modes [36]. The bidirectional LSTM has shown to have recorded higher recognition accuracies over the other LSTM models [37]. However, LSTMs are computationally intensive and sometimes the gradient decay is highly dominant due to *tanh* function that becomes hard to ignore. The solution to the above problems came in the form a new improved architecture called as independent recurrent neural networks. These models were able to develop longer and deeper architectures without vanishing gradients problem [36]. However, it is implied that recurrent models were indecisive on spatial features which defined the joint relationships with in a skeletal action frame. Hence, spatial temporal combination networks were proposed with CNNs followed by LSTMs [38]–[40] for action recognition. The CNNs learned spatial joint features and the flattened features in the dense layers of CNN are inputted to LSTMs to determine temporal patterns in the extracted spatial contents. Despite higher recognition accuracies, the CNN LSTM models are not end-to-end trainable in most of the action recognition framework proposed in literature [38].

To overcome these network implications for action recognition, a rich spatio temporal feature representations in the form of RGB color images. These RGB color maps characterize a particular skeletal action across a set of 3D video frames. Consequently, the proposed spatio temporal images are found to be independent of length of the video sequences as well as number of joints. These spatio temporal features represent spatial relationships among joints within a 3D action frame and temporal changes between frames as we move horizontally representing temporal patterns. The proposed spatio temporal features are joint positional maps (JPM) [41], Joint Distance Maps (JDM) [11], joint Angular maps (JAM) [42], Joint Angular displacement maps (JADM) [17], Joint Velocity maps (JVM) [43], Joint acceleration maps (JaM) [44], joint planar maps (JpM) [45], joint trajectory maps (JTM) [46] and quad joint volume maps (QJVM). The above spatio temporal feature maps are embedded with patterns that can be quantified using a deep CNN of any architecture. It has been shown that the deep CNNs had certainly enhanced the performance of the skeletal action recognition system on Kinect and mocap datasets.

Undoubtedly, the above analysis shows that the spatio temporal feature maps can be learned exceptionally well by the deep networks. But, what if the joint orders on the skeleton changes during the experimentation across datasets and machines. Following the discussion, we propose to investigate, Can Skeletal Joint Positional Ordering Influence Action Recognition on CNNs for Achieving Joint Order Independent Learning. Fig.1 and 2 show how joint order independence is necessary if multiple datasets are being used for testing a proposed skeletal based action recognition system. We also found evidences where the independent researchers used different joint orderings on the same datasets [12]–[15]. Surprisingly, we found joint orders play a crucial role in evaluating classifier models for skeletal action recognition. Hence, the outcome of this paper which answers the question: can we achieve joint order independence on deep networks, is threefold. The first one being the design of spatio temporal feature maps to represent jumbled skeletal joints that can achieve order independence. Secondly, the number of these randomly ordered training feature maps necessary to develop a reasonably accurate classifier.

Thirdly, to find a deep learning architecture that will guarantee highest recognition accuracy on Jumbled maps of certain feature type. We demonstrate the entire process through experimentation. The following section illustrates the underlying methodology for the proposed hypothesis and its evaluation.

III. METHODOLOGY

Skeletal action recognition using deep learning models has to attain a remarkable level of flexibility in disregarding skeleton joint orders during feature computation. Despite a large contingent of these successful methods have been proposed on skeletal based action recognition, this is the first time to report the effects of joint order variations on their performance. There are three challenges in designing a universal deep learning system for skeletal action recognition. One, to choose how many randomly generated joint ordering feature maps are required for training, what should be the optimal CNN architecture for achieving joint independent learning and finally, to investigate which type of features give optimized performance with high accuracies. Here, we set the procedures for creating various feature maps along with our novel quad joint volumetric features, generating random joints given a joint order and designing a Spectrally Graded CNN architecture. This section consists of six subsections: describing the extraction of features from joints and converting to maps; our proposed QJVM features; generating random joint orders; the proposed SgCNN; Its training; testing and evaluation.

A. SPATIO TEMPORAL FEATURE MAPS

The spatio temporal features define a human skeleton joint's inter and intra frame relationships across 3D action video frames. The human skeleton is represented digitally with J joints which convey their spatial location with respect to the camera coordinates. These spatial locations are positional vectors defined as $p_J = (x_i, y_i, z_i) \in R^{3\times J} \ \forall \ i = 1 \text{ to } J$. Currently, all the features are extracted from the positional vectors which provide spatial temporal relationships between the joints during an action. The first methods converted the joint positions in x into a red (R) colour coded plane, y into a green (G) and z into a blue (B) using a threshold on each of these positional values [41]. We call these coloured positional feature maps as Joint Positional Maps (JPM). Similarly, the method in [47] converts $x-y$, $y-z$ and $z-x$planes into R, G and B planes to create action feature maps, which are called as Joint Paired Positional Maps (JPPM). The above two methods applied three stream CNNs with 8 convolutional layers with max pooling and ReLu operations in between them. Each stream has a dense layer followed by a SoftMax layer. The output class probabilities are predicted using a decision score fusion model to recognize actions. The results are better than the previous non deep machine learning models due to multi feature learning which was automated in deep learning models. However, the recognition score was further improved through a small modification to the model in [48] by adding a 4th stream of xyz combined feature map from [11] in [49].

The above methods propose spatio temporal maps that does not explore relationships among the joints in spatial and temporal domains. This was achieved using joint distance maps (JDMs) in [11] through joint pairs. For a J joint 3D skeleton, there are ${}^{J}C_2$ joint combinations accounting for $\frac{J(J-1)}{2}$ unique pairs. In 3D video frame t, the paired $i^{th}$and j^{th} joints represented by positional pointers $p_i = (x_i, y_i, z_i)$ and $p_j = (x_j, y_j, z_j)$ respectively, develop a l_2 norm based relational features expressed as

$$d_{ij} = \left\| p_i - p_j \right\|_2 \tag{1}$$

where, d_{ij} is the Euclidian distance between two joints in a frame. For the entire frame with J joints, d_{ij} becomes d_J^t. The d_J^t is a vector describing all the joint relationships within a frame t. Extending on to the entire skeletal 3D video action sequence with T frames, we represent d_J^t as a matrix $d_{\frac{J(J-1)}{2}}^T$ of size $\frac{J(J-1)}{2} \times T$. Hence for all three axis, we have a distance feature matrix of size $\frac{J(J-1)}{2} \times T \times 3$. In [17], these three planes are colour coded into RGB planes to form a feature map that represents the spatio temporal variations in the 3D skeletal action sequence. The maps created were called as joint distance maps (JDMs). The performance of JDMs was found to be better on a single stream CNN network which is less complex and computationally efficient than the networks using in [40], [50] and [51].

The JDMs were further enhanced by joint angular information into a more robust feature representation through joint angular displacement maps (JADMs) [17]. The JADMs are created by combining joint distance features with their orientation angular information. The angular displacement features between the i^{th} and j^{th} joint in a t^{th} frame is formulated as

$$d_{\angle ij}^t = d_{ij}^t \times \cos\left(\theta_{ij}^t\right) \tag{2}$$

where

$$\theta_{ij}^t = \cos^{-1}\left(\frac{\vec{P_{ik}}\,\vec{P_{kj}}}{\sqrt{\vec{P_{ik}}}\sqrt{\vec{P_{kj}}}}\right) \forall t \tag{3}$$

The orientation angle θ_{ij}^t in each frame t is a vector which is computed with respect to a common adjacent joint k. The $\vec{P_{ik}} = d\,(i,k) \in R^3$ and $\vec{P_{kj}} = d\,(k,j) \in R^3$ are joint projection vectors. Consequently, it transforms into a $\frac{J(J-1)}{2} \times T$ feature matrix which represents a 3D action video sequence. The obtained feature matrix in 3D is colour coded to form a $\frac{J(J-1)}{2} \times T \times 3$ RGB image. The JADMs characterized very subtle joint variations across 3D actions, thus transforming robust patterns into the image pixel representations that provided good discriminations across actions.

Alternatively, enhanced feature maps such as joint velocity maps(JVMs) [43], joint angular maps (JAMs) [42], joint angular velocity maps(JAVMs) [52], joint trajectory maps(JTMs) [46] and joint acceleration maps(JaMs) [44] with deep learning networks have shown to improve recognition accuracies over traditional features. All these maps model 2-joint relationships within a 3D video action sequence. Substituting 2-joint with 3-joint relationships has further enriched the patterns on the maps for automated feature extraction process in CNNs. The 3-joint relational feature maps were called joint planar maps (JpMs) [45] and joint

surface maps (JSMs) [53]. Inspired, we propose a 4-joint relational map called as quad joint volume feature map (QJVMs) which is elaborated in the following section.

However, we discovered that the created feature maps are based on one simple rule: Never change the skeletal joint ordering. Changing joint ordering during feature computation process across experiments greatly affects the performance of the deep learning algorithms. Extensive experimentation and analysis has been performed in this work to meet the proposed objective of discovering a universal action recognition framework which is independent of the joint ordering.

B. QUAD JOINT VOLUME FEATURES

The human skeletal model is represented on a machine with J joints which forms ${}^{J}C_4$ unique 4-joint pairs. In 3D space, each joint is represented as a position vector described by $p_J = (x_i, y_i, z_i) \in R^{3\times J} \ \forall \ i = 1 \text{ to } J$. To construct a geometric quadrilateral with 4 sides, we use the ${}^{J}C_4$ four joint pairs. Hence, on a J joint skeleton, we construct ${}^{J}C_4$ quadrilaterals in 3D space from. For our 39-joint action skeleton, we construct 82251 four-sided 3D quadrilaterals of arbitrary shapes. These 82251 polygons characterize all possible relations among joints in a 3D frame t. Joint volume features of the 82251 quadrilaterals describe the spatial joint relationships within a 3D video action frame. However, we eliminated the slow varying quadrilaterals across action frames using averaging threshold. Hence, only 10% of 82251 have impactful quadrilaterals in each action sequence, which are useful for feature computation.

During skeletal motion in the 3D video sequence, the constructed 3D quadrilaterals vary shapes and orientations proportionally with respect to the joint relationships. Thus, transforming these changes into quad joint volume feature (QJVF) matrices. To find the volume of the any 3D quadrilateral described by the coordinates $\{(x_i, y_i, z_i), (x_{i+1}, y_{i+1}, z_{i+1}), (x_{i+2}, y_{i+2}, z_{i+2}), (x_{i+3}, y_{i+3}, z_{i+3})\}$ of its vertices are known, we apply the following process to calculate quad joint volume. Fig.3 shows the process of designing relative quad joint volume features. To find the volume of the irregular quadrilaterals, we split it into two truncated triangular prisms and we find the volume using the expression

$$V_s^t = \frac{z_i + z_{i+1} + z_{i+2}}{6} \begin{vmatrix} 1 & x_i & y_i \\ 1 & x_{i+1} & y_{i+1} \\ 1 & x_{i+2} & y_{i+2} \end{vmatrix} + \frac{z_i + z_{i+2} + z_{i+3}}{6} \begin{vmatrix} 1 & x_i & y_i \\ 1 & x_{i+2} & y_{i+2} \\ 1 & x_{i+3} & y_{i+3} \end{vmatrix} \quad (4)$$

where (x_i, y_i, z_i) is the starting coordinate of the s^{th} quadrilateral in the t^{th} frame. Hence the QJVF is a vector of size $0.1 \times {}^{J}C_4 \times 1$ representing 3D quadrilaterals volume. For the entire 3D video sequence with T frames, the spatio temporal QJVF is a matrix of size $0.1 \times {}^{J}C_4 \times T \in R^2$. Finally, for a dataset with N labelled 3D videos, the QJVF is a multidimensional matrix of size $0.1 \times {}^{J}C_4 \times T \times N$.

Finally, for a dataset with N labelled 3D videos, the QJVF is a multidimensional matrix of size $0.1 \times {}^{J}C_4 \times T \times N$. The chronological arrangement of these features is shown in Fig.3.

In general, the above process is expandable on skeletal data captured using sensors like Kinect or a mocap system with different camera setup other than the one used in this work. Therefore, we proceed to investigate the performance of QJVFs on publicly available HDM05 [12] and CMU mocap [13] and NTU RGB D [14] Kinect dataset along with our own 3D mocap dataset KLHA3D-102. The QJVF feature matrix can be used for training the classifiers directly or can be encoded as color images for training on deep convolutional neural networks. Despite the success of classifiers like HMM and DTW on such time series data, their operating efficiency reduces on large datasets, such as the one used in this work. Hence, we encode the QJVFs into color coded pixels using the procedures from our earlier work [17]. The volume data is color coded into RGB planes using the 'jet' color map to form quad joint volume (QJVMs) feature maps, by following standard mapping procedure [54],

C. JUMBLE SKELETAL JOINTS

The color coded feature maps represent joint variations in 3D skeletal actions as pixels on an image. These pixel patterns are learned by deep CNN models for recognition of human actions across classes. The studies on these feature maps have revealed two interesting observations.

1) The joint ordering is fixed at the start of the experiment by the capturing sensors, which can be modified by the user based on the missing joint information after recording.
2) Training and testing with different joint orders hasn't been conducted on the deep learning models to understand its implications on overall performance of the skeletal based action recognition systems.

Hence, this work proposes to explore the impact and importance of joint ordering in skeletal based human action recognition tasks from feature maps to training a deep network.

The primary goal is to create a jumbled joint ordering, given the sensor generated skeletal joints. The initial skeletal joint ordering in our motion captured KLHA3D-102 is shown in fig.1(e). This is the joint ordering that was selected during data capture. To create a different ordering of joints, we used "The Fisher–Yates shuffle" algorithm [55]. The function shuffles the inputted J joint list randomly to produce a different joint ordering. In this work, the shuffle routine was called 100 times with the original J joint input in every instance. Out of 300, we selected 100 joint orders that were found to be uniquely random through a cross correlation coefficient on the original and the generated joints. Specifically, the selected 100 joints are highly discriminating and independent orders based on expectation minimization of correlation coefficient. Hence, we selected these 100 shuffled joint orders for training and testing the proposed problem through deep networks.

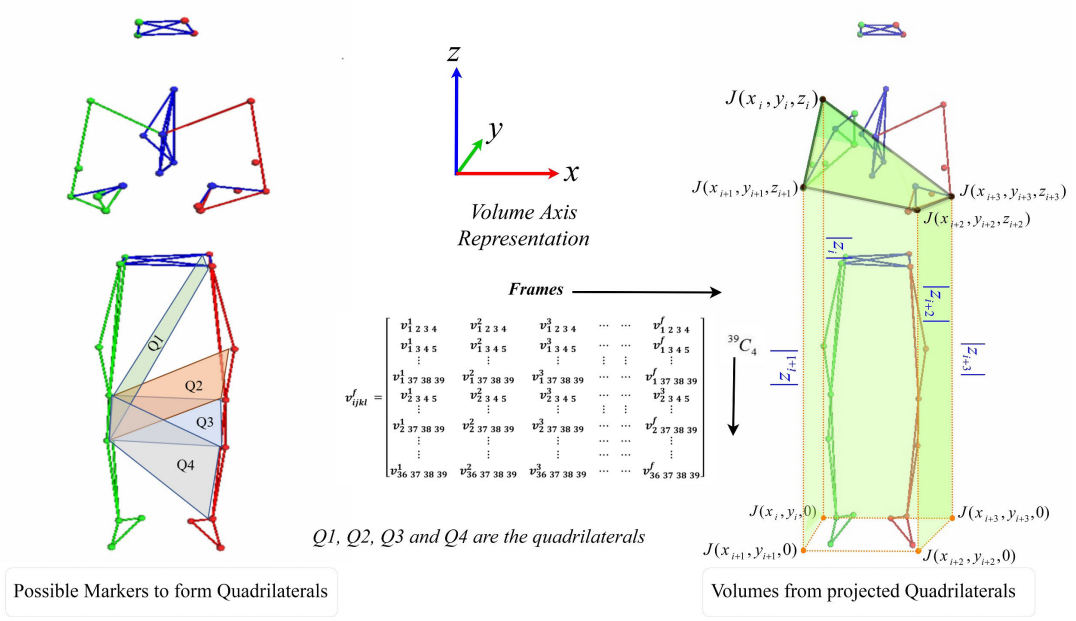

FIGURE 3. Quad joint volume features (QJVF).

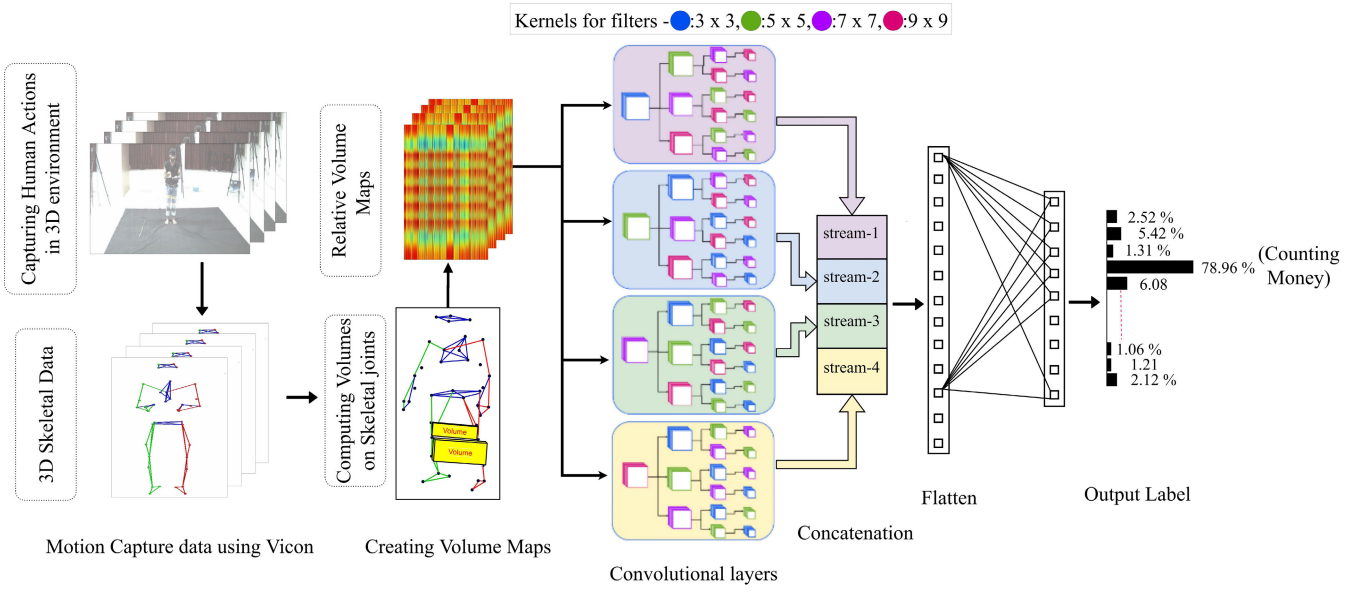

FIGURE 4. Spectrally enriched graded convolutional neural network (SgCNN) using QJVMs for action recognition.

Fig.1 shows the projected shuffled joints onto the human skeleton.

However, repeating the same joint orders is not possible as they are generated randomly. Hence, we performed the experiments 5 times from scratch on different DL frameworks and across action dataset features. Consequently, testing is initiated to find the number of random joint order training set necessary to achieve good recognition accuracies.

The methods for creating feature maps were initiated on these 100 joint orders. The fig.5(a) shows the joint angular displacement maps (JADM) on the 10 jumbled joint orders. Similarly fig's.5(b) to 5(c) show feature maps of joint distance maps (JDM) and quad joint volume maps (QJVM) used in this work which are computed from the 100 jumbled joints.

In addition to our motion capture dataset, we experimented to discover universality of the proposed framework across benchmark skeleton action datasets such as NTU RGB D [14], MSRACTION3D [15], HDM05 [12] and CMU [13]. The HDM05 and CMU are recorded on 3D motion capture platform, whereas the others are based on Kinect sensor.

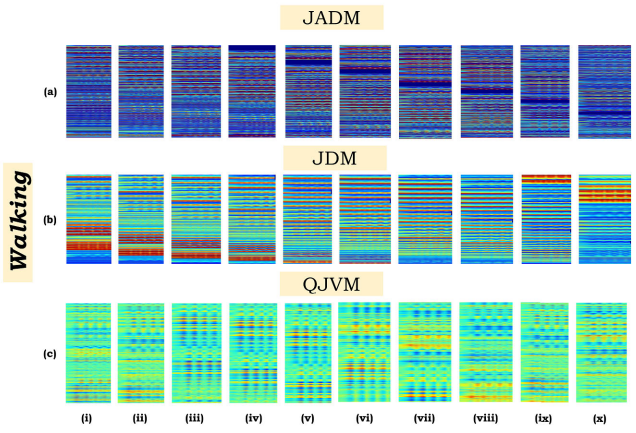

FIGURE 5. Three feature maps from our previous works for 10 jumbled joints.

Joint shuffling and spatio temporal feature map creation are consistent across datasets. The existing deep networks such as CNN, ResNets, GoogleNet and recurrent CNN have learned the above feature maps with a fairly small training loss but couldn't generalize during validation. The validation errors have become constant after 50 epochs for most of the feature maps across datasets. Hence to improve the recognition accuracies, we built a grid-like circular transferable feature model that we call a spectrally enriched graded CNN (SgCNN), the architecture of which is discussed in detail in the next section.. The following section describes SgCNN architecture, training and testing procedures.

D. SPECTRALLY GRADED CNN

The aim of Spectrally Graded CNN (SgCNN) is to rotate the multidimensional features around the network to create a nonlinear feature vector that can generalize from the input data samples. This approach is used by ResNets and RNNs to reinforce lost data and avoid vanishing gradients. These networks are very deep; the minimum number of layers has been found to be around 20. Our SgCNN differs from existing networks in three respects.

1) It processes multiband features simultaneously to generate a highly nonlinear feature vector and thus avoid over- or underfitting.
2) It does not use a dropout layer to induce random feature nonlinearity before the dense layer.
3) It is computationally efficient and requires less training.

The SgCNN architecture is shown in fig 4.

Before applying the proposed SgCNN to QjRVMs and other maps, we needed to evaluate its performance on standard image datasets. For this, we used several image and video datasets, namely Fruits-360 [56], Food-101 [57], Caltech-256 Object Categories [58], KTH-Animal [59], and UCF Sports [60]. These were used to compare the proposed SgCNN against state-of-the-art network architectures such as CNN8 [22], VGG16 [61], VGG19 [62], ResNet-50 [63], GoogleNet [64], SENet-154 [65] and proposed SgCNN. The results of experimentation were presented in table 1. The results induced confidence on the SgCNN architectures ability to learn multiple resolution patterns simultaneously in images. The next subsection describes training of the SgCNN on randomly shuffled skeletal joint feature maps.

E. TRAINING SgCNN

To implement the SgCNN algorithm, we used Python 3.6, with TensorFlow library to train the model. We used the same hyperparameters for all datasets, except for the learning rate, which was reassigned during training for each dataset. Specifically, we decreased the learning rate exponentially from 0.01 until the error became constant. At the start of the training phase for each dataset, we set the network's weights and bias parameters randomly using a zero-mean Gaussian distribution function with variance 0.01.

The SgCNN learned by updating its weights and bias parameters using the back propagation gradient descent algorithms. Consequently, to contain the validation errors during training we applied a l2 weight regularizer after each convolutional layer.This has enabled the SgCNN to develop uniformity in weights across layers during training. We applied ReLU and SoftMax hyperparameter activations in the convolutional and dense layers, respectively. Finally, we used a fixed batch size of 32 for training, based on the image resolution and amount of GPU memory available. The training was performed on a NVIDIA GTX 1070, 8GB GPU with the model. The SgCNN model is built from scratch with keras frontend and tensorflow backend.

F. TESTING AND EVALUATION

Testing sets of different proportions were etched out using multiple combination of spatio temporal feature maps constructed by mitigating skeletal joint orderings. Here, we find solution to our third objective which aims to find the number of optimal joint order combinations required to achieve joint order independence. Consequently, testing process has been exhaustive which included multiple instances of executions across combinations of maps in all the considered datasets. The performance of the SgCNNs was evaluated based on recognition accuracies averaged across datasets.

IV. EXPERIMENTATION AND ANALYSIS

Exhaustive experiments were designed and subsequently conducted to discover spatio temporal feature maps and their train test ratios on deep networks that are better suited to deal with the skeletal joint order variations across datasets for action recognition tasks. We start by evaluating QJVMs on SgCNN against various feature maps and deep networks with base joint order on KLHA3D102 dataset. The base joint order is the initial joint orientation followed in the action datasets. Jumbled joint order is the randomly shuffled skeletal joints using ''The Fisher-Yates Shuffle''. The obtained results were then compared against benchmark datasets. Next, the above experiments are repeated for jumbled joints to identify the type of skeletal feature maps that will achieve joint order independence through optimal training on different network

architectures. Finally, the random joint shuffle routine is called 5 times on 5 differently configured computer systems to generate 25 randomly shuffled skeletal joint ordered maps per class for testing and identify best in class deep networks that allow joint independent learning.

A. SKELETAL ACTION DATASETS

The KLHA3D102 is captured with an 8 camera vicon motion capture technology [18]. The human skeleton in our dataset has 39 joints from head to toe. The joints in 3D mocap are placed manually by pre- determining the highly articulated joints on the human body. In total KLHA3D102 consists of 102 classes with 10 subjects and each repeating the action 10 times. Hence we have, $102 \times 10 \times 10 = 10200$ skeletal actions with a base joint order representation. In order to achieve a skeletal joint independent action recognition model, we now consider enumerating the base joint order to multiple random jumbled joint orders. As discussed in section III-C, a 100 shuffled joint ordered skeletons were generated. Subsequently, associated action datasets were formulated from these jumbled skeletons. Finally, features maps were extracted on these jumbled joint datasets. The complete jumbled joint dataset consists for a particular feature type has a set of $102 \times 10 \times 10 \times 100 = 1020K$ feature maps. The size of feature map images are fixed to 256×256 for optimum loading effect on the GPU during training. Similarly, the above process is repeated for creating maps across 10 feature types as discussed in the section III-A.

Identically, we followed the above procedure to create jumble joint dataset for our KLYOGA3D. This is a 42 class yoga skeletal action dataset recorded with 10 subjects and 5 repetitions. The total size of jumbled joint feature maps on the yoga dataset would be $42 \times 10 \times 5 \times 100 = 210$K per feature type. For all 10 feature types combined, we have 10200K on KLHA3D-102 and 2100K on KLYOGA3D respectively.

Apart from our KLHA3D102, we also used publicly available 3D mocap action datasets HDM05 and CMU. Out of the two, HDM05 was less noisy and consists of 70 action classes with 5 subjects performing an action several times. In this work, we used $70 \times 30 \times 5 \times 100 \times 10 = 1575000$ action samples for training and testing. The CMU dataset in this work is carefully crafted to avoid missing and noisy marker information. Consequently, the CMU dataset used for training and testing has $30 \times 30 \times 10 \times 100 \times 10 = 9000000$ samples, with 30 actions classes, 10 subjects and 30 variations per subject. On the contrary to our 39-joint skeleton, HDM05 and CMU are captured with 41-joint skeletons. Finally, to discover the usefulness of the proposed maps and the ML algorithm, we investigated Kinect skeletal action data with 25 joints from NTU RGB D dataset. Our refabricated NTU RGB D dataset has $60 \times 30 \times 10 \times 100 \times 10 = 1800000$ action samples. Besides, these datasets were selected to have a 30 to 40% overlap among action classes.

The entire experimentation is divided into 3 clusters in which different experiments will be conducted. In the first cluster (C1) we test our proposed feature maps QJVF and the SgCNN architecture across multiple features on considered datasets. In C1, only base joint order or any random joint order skeleton is used and the joint order is fixed throughout the experimentation. Experiments in C1 test the usefulness of our proposed spatio temporal feature maps QJVFs against the existing maps on deep networks.

The second cluster (C2) is what makes this work really interesting. Here, we train deep networks to predict human actions with random joint ordered skeletal maps. We created a total of 100 random jumbled ordered joint feature maps per action per subject per repetition across datasets. The entire dataset has been divided into multiple train and test samples of different dimensions to discover the necessary train test ratios for joint independence in skeletal action recognition systems across datasets. We recorded the end-to-end system accuracies over a multitude of these train test ratios and discovered a possible range for joint order independent learning by the deep learning models. Additionally, we also recognized the best of joint feature maps that are suitable for achieving our proposed objective.

Finally, cluster C3, is designed to validate the proposed jumble joint independent learning across multiple machines. This phase is necessary to ensure that the required number of training samples doesn't change by a large margin across different hardware configurations. The proposed method generates the joint orders randomly which fluctuate between experiments. Therefore, we performed the experiments 5 times on 5 different hardware configurations to determine the possibility of a joint order independent learning for skeletal based action recognition.

B. C1: MONOSKEL RESULTS

To begin with, we focus on the performance of our proposed QJVMs and the novel architecture SgCNN as a traditional approach where the skeletal joints are unaltered throughout the experiment, MONOSKEL. The focus will be to test the effectiveness of quad joint relationships for skeletal based action recognition on deep learning architectures. Also, test the performance of the proposed SgCNN for classification tasks. We compare and analyze the test results with respect to different state-of-the-art maps and networks for skeletal action recognition.

1) EVALUATING QJVM AND SgCNN ON KLHA3D102

The performance metrics for evaluation is maintained uniformly across the work as mean recognition accuracy. Here, we divided the entire KLHA3D102 dataset into multiple training units of different train test ratios. Specifically, we will identify MONOSKEL Accuracy Maximization Samplers (AMS) on the training data. However, the AMS can be emphasized as the minimum amount of training samples necessary for generating maximum recognition accuracy. After many different iterations, we selected to start at 20 and reach up to 80 training samples per class with an increment of 20 samples. Hence, a specific range is being discovered as AMS. The trained networks are tested with

TABLE 1. Prediction accuracies achieved for all test sets.

Datasets	Recognition Rates (%)						
	CNN8 [22]	VGG16 [61]	VGG19 [62]	ResNet-50 [63]	GoogLeNet [64]	SENet-154 [65]	SgCNN (Proposed)
Fruits-360 [56]	86.15	83.74	85.24	85.64	86.16	87.24	97.24
Food-101 [57]	76.24	62.31	67.68	68.22	72.18	78.16	88.45
Objects-256 [58]	78.95	72.76	71.35	69.58	75.31	80.45	92.91
KTH-Animals [59]	81.24	74.81	76.14	75.19	80.62	84.22	94.06
UCF Sports [60]	82.54	76.17	77.59	79.64	81.24	83.47	93.49

TABLE 2. Recorded accuracies of different features maps on State of the art models.

Training Samples	Feature Maps	CNN8 [22]	VGG16 [61]	VGG19 [62]	ResNet-50 [63]	GoogLeNet [64]	SENet-154 [65]	SgCNN (Proposed)
20	JPM [41]	49.47	52.58	54.40	55.19	56.31	56.80	88.31
	JDM [11]	54.50	57.37	60.34	62.66	64.00	66.17	66.77
	JAM [42]	51.74	54.61	57.58	59.90	61.24	63.41	64.01
	JADM [17]	67.89	69.54	70.45	73.04	74.54	76.34	76.42
	JSM [53]	54.05	56.87	59.52	59.64	70.87	61.24	61.32
	JVM [43]	64.23	66.96	68.14	70.32	72.05	74.20	74.52
	JaM [44]	64.80	67.46	68.27	70.65	71.88	74.58	74.88
	JpM [45]	66.36	68.01	68.92	71.51	73.01	74.81	75.52
	JTM [46]	49.78	52.60	55.25	55.37	56.60	56.97	57.05
	QJVM	68.57	70.22	71.13	73.72	75.22	75.36	77.02
40	JPM [41]	59.16	62.11	63.93	64.72	65.84	66.33	67.84
	JDM [11]	65.14	68.01	70.98	73.30	74.64	76.81	77.41
	JAM [42]	63.31	66.18	69.15	71.47	72.81	74.98	75.58
	JADM [17]	79.14	80.79	81.70	84.29	85.79	87.59	87.67
	JSM [53]	75.30	78.12	80.77	80.89	82.12	82.49	82.57
	JVM [43]	75.48	78.21	79.39	81.57	83.30	85.45	85.77
	JaM [44]	76.05	78.71	79.52	81.90	83.13	85.83	86.13
	JpM [45]	77.61	79.26	80.17	82.76	84.26	86.06	86.77
	JTM [46]	71.03	73.85	76.50	76.62	77.85	78.22	78.30
	QJVM	79.82	81.47	82.38	84.97	86.47	86.32	88.27
60	JPM [41]	73.13	76.24	78.06	78.85	79.97	80.46	81.97
	JDM [11]	69.27	72.14	75.11	77.43	78.77	80.94	81.54
	JAM [42]	67.44	70.31	73.28	75.6	76.94	79.11	79.71
	JADM [17]	83.27	84.92	85.83	88.42	89.92	91.72	91.78
	JSM [53]	79.43	82.25	84.9	85.02	86.25	86.62	86.45
	JVM [43]	79.61	82.34	83.52	85.7	87.43	89.58	89.9
	JaM [44]	80.18	82.84	83.65	86.03	87.26	89.96	90.26
	JpM [45]	81.74	83.39	84.3	86.89	88.39	90.19	90.9
	JTM [46]	75.16	77.98	80.63	80.75	81.98	82.35	82.43
	QJVM	83.95	85.6	86.51	89.1	90.6	91.25	92.53
80	JPM [41]	75.31	78.42	80.24	81.03	82.15	82.64	84.15
	JDM [11]	72.5	75.37	78.34	80.66	82.16	84.17	84.77
	JAM [42]	70.67	73.54	76.51	78.83	80.17	82.34	82.94
	JADM [17]	86.5	88.15	89.06	91.65	93.15	94.95	95.03
	JSM [53]	81.61	84.43	87.08	87.2	88.43	88.58	88.88
	JVM [43]	83.41	86.14	87.32	89.50	91.23	93.38	93.70
	JaM [44]	83.98	86.64	87.45	89.83	91.06	93.76	94.06
	JpM [45]	85.54	87.19	88.10	90.69	92.19	93.99	94.70
	JTM [46]	77.34	80.16	82.81	82.93	84.16	84.53	84.61
	QJVM	88.16	89.81	90.72	93.31	94.81	95.38	96.61
Memory Usage		315MiB	396MiB	415MiB	284MiB	297MiB	308MiB	236MiB
No. of Parameters		24.9M	134.6M	140.2M	9.4M	10.6M	23.6M	10.2M

the remaining 20 samples. Moreover, 30% of the training data is applied for validation during training sessions. Therefore, there are 4 training sessions per dataset. Notably, it is not possible to maintain hyper parameter consistency across different network architectures that are used in this work. However, we maintained, weight and bias initialization along with learning rates as constant for all networks used in this work. Stochastic gradient descent optimization was used for updating the trainable deep parameters. The learning rate decays are activated by 10% when the loss appeared to be constant for 10 epochs and training is stopped if loss doesn't decay for 20 epochs during training.

TABLE 3. Performance evaluation of QJVM and SgCNN on benchmark dataset.

Performance evaluation of QJVM and gCNN on benchmark dataset							
Datasets	CNN8 [22]	VGG16 [61]	VGG19 [62]	ResNet-50 [63]	GoogLeNet [64]	SENet-154 [65]	SgCNN (Proposed)
HDM05 [12]	78.01	79.61	80.08	80.23	82.35	85.76	87.46
CMU [13]	75.09	76.28	77.12	78.40	80.15	83.47	85.31
MSR Action3D [15]	78.27	82.14	81.56	82.43	85.50	86.77	89.20
NTU RGB D [14]	82.18	84.52	85.21	86.02	87.51	90.72	92.21
KLYoga3D [17]	83.89	86.48	87.37	88.10	91.44	93.51	95.48
KLHA3D-102 [16]	85.53	86.52	87.64	88.08	90.51	92.85	93.82

TABLE 4. Performance evaluation of QJVM and SgCNN on KLHA3D-102.

Evaluating QJVM and gCNN on KLHA3D-102								
Training Samples	Feature Maps	CNN8 [22]	VGG16 [61]	VGG19 [62]	ResNet-50 [63]	GoogLeNet [64]	SENet-154 [65]	SgCNN (Proposed)
20	JPM	7.14	10.25	12.07	12.86	13.98	14.47	15.98
	JDM	45.61	48.48	51.45	53.77	55.11	57.28	57.88
	JAM	42.85	45.72	48.69	51.01	52.35	54.52	55.12
	JADM	59.00	60.65	61.56	64.15	65.65	67.45	67.53
	JSM	9.73	12.55	15.20	15.32	16.55	16.92	17.00
	JVM	55.59	58.32	59.50	61.68	63.41	65.56	65.88
	JaM	56.16	58.82	59.63	62.01	63.24	65.94	66.24
	JpM	57.72	59.37	60.28	62.87	64.37	66.17	66.88
	JTM	5.46	8.28	10.93	11.05	12.28	12.65	12.73
	QJVM	59.73	61.38	62.29	64.88	66.38	72.50	68.18
40	JPM	16.83	19.78	21.60	22.39	23.51	24.00	25.51
	JDM	56.25	59.12	62.09	64.41	65.75	67.92	68.52
	JAM	54.42	57.29	60.26	62.58	63.92	66.09	66.69
	JADM	70.25	71.90	72.81	75.40	76.90	78.70	78.78
	JSM	20.98	23.80	26.45	26.57	27.80	28.17	28.25
	JVM	66.84	69.57	70.75	72.93	74.66	76.81	77.13
	JaM	67.41	70.07	70.88	73.26	74.49	77.19	77.49
	JpM	68.97	70.62	71.53	74.12	75.62	77.42	78.13
	JTM	16.71	19.53	22.18	22.30	23.53	23.90	23.98
	QJVM	70.98	72.63	73.54	76.13	77.63	83.75	79.43
60	JPM	20.80	23.91	25.73	26.52	27.64	28.13	29.64
	JDM	60.38	63.25	66.22	68.54	69.88	72.05	72.65
	JAM	58.55	61.42	64.39	66.71	68.05	70.22	70.82
	JADM	74.38	76.03	76.94	79.53	81.03	82.83	82.89
	JSM	25.11	27.93	30.58	30.70	31.93	32.30	32.13
	JVM	70.97	73.70	74.88	77.06	78.79	80.94	81.26
	JaM	71.54	74.20	75.01	77.39	78.62	81.32	81.62
	JpM	73.10	74.75	75.66	78.25	79.75	81.55	82.26
	JTM	20.84	23.66	26.31	26.43	27.66	28.03	28.11
	QJVM	75.11	76.76	77.67	80.26	81.76	86.73	83.69
80	JPM	22.98	26.09	27.91	28.70	29.82	30.52	31.82
	JDM	63.61	66.48	69.45	71.77	73.27	75.28	75.88
	JAM	61.78	64.28	67.62	69.94	71.28	73.45	74.05
	JADM	77.61	79.26	80.17	82.76	84.26	86.06	86.14
	JSM	27.29	30.11	32.76	32.88	34.11	34.48	34.56
	JVM	74.77	77.50	78.65	80.86	82.59	84.74	85.06
	JaM	75.34	78.00	78.81	81.19	82.42	85.12	85.42
	JpM	76.90	78.55	79.46	82.05	83.55	85.36	86.06
	JTM	23.02	25.84	28.49	28.61	29.84	30.21	30.29
	QJVM	78.19	79.83	80.41	83.46	84.38	84.87	86.15

The state-of-the art networks such as CNN8 [22], VGG16 [61], VGG19 [62], RESNET-50 [63], GoogleNet [64], SENet-154 [65] which were highly competitive during ImageNet classification challenges are being considered for validating the proposed SgCNN. Table 2 gives the entire results of experimentation in C1. The proposed QJVMs consider 4 - joint combinations to calculate features instead of 2 or 3. Although, 4 - joint features need a large computational space, it is relatively richer in characterizing joint dependencies across actions. This is similar to 2 or 3 joint features

except for the fact that the 4 joint features are extracted from the closed 3D volume between joint spaces. Hence, the feature maps QJVMs show high pixel patterns that are necessary for discriminating closely related actions. Hence, the accuracy on all the state-of-the-art CNNs for image classification problems have shown to generalize well on the proposed QJVMs. However, the most unsuccessful is the joint positional maps (JPMs) due to non discriminating pixel patterns between closely related actions. We can also observe that the maps with differentiating features such as JVM and JaM have produced good recognition accuracies. Similarly, higher dimensional features such as JpMs and JADMs are not far behind differentiating features. Overall, we found that a highly relational feature on the skeletal joints has provided good action recognition capabilities.

Secondly, the deep networks that were used in this work have already proved their might in the image classification space. However, the amount of data used for training these models is quite different from their usual datasets. All the models were trained from scratch on a 8GB GPU, GTX1070 with the same initial hyper parameters. Table 2 shows the recorded accuracies on different features maps. Since all the networks used are state-of-the-art, their accuracies across maps didn't have large margin. Interestingly our proposed SgCNN has proved to be competitive along with these models. However, what separates the state-of-the-art from SgCNN is the computational complexity and memory usage during training, which are tabulated in last two rows of table 2. In particular, these last two rows show that the proposed network has less trainable parameters and occupies less memory making it stand out among the best. The reason for this would be the parallel architecture and gyroscopic filter Kernels that facilitate hyper hierarchical feature representation across multiple channels. Moreover, higher resolution filter Kernels above 9 has found to have little improvement in recognition accuracies on images and hence 9 was the maximum size considered for SgCNN.

Finally, the first column in table 2 shows the amount of training maps considered per feature per deep network. This is to identify the AMS necessary to achieve prediction confidence of the network. From table 2 AMS can be the range of 60 to 80 training samples of MONOSKEL data. However, AMS is subjective to operating GPU systems. Alternatively, we tested on 4 other GPU configurations and found that there was around ± 3 % variation in accuracy levels. To summarize, the optimal value of AMS for our action dataset ranges between 60 to 80 samples per class when the skeletal joint orders are unchanged during training and testing periods.

2) MONOSKEL ACROSS BENCHMARK DATASETS

To validate the proposed framework for skeletal action recognition with respect to different data sources, we applied the benchmark datasets from KLYOGA3D, NTU RGB D, UTKINECT, MSRACTION3D, HDM05 and CMU. Since each of these datasets were discussed elaborately in the start of this section, we present the results in table 3. Moreover, the training and testing samples differ in each case as they are unevenness in the number of classes and the number of images per class. Here the intuition is to test performance of the proposed framework and not to pick the best possible solution for action recognition. Table 3 generates confidence in our proposed framework through the computed mean recognition accuracies that are close to normal. However, the accuracies in table 3 are on the lower side when compared to table 2, due to noisy datasets except ours KLYOGA3D. Illustrating on table 3 allows us to contribute a novel interface for skeletal action recognition. In the following section, we present cluster C2, where the networks are taught to learn features from jumbled skeletal joints.

C. C2: JUMBLESKEL RESULTS

In this cluster, we present the results of skeletal action recognition tasks using features constructed using jumbled joints on deep networks. This cluster is the most captivating part of the entire experimentation. The focus would be to discover the JUMBLESKEL AMS on a particular set of features. We also extend this by analyzing networks on which a maximum accuracy is achievable. Finally, the results of JUMBLESKEL features on different skeleton sources when they interact with the deep network.

1) EVALUATING JUMBLESKEL ON KLHA3D102 DATASET

This 1020000 jumbled joint feature maps consists of all actions from different subjects with multiple orientations. Undoubtedly, one of the objectives is to find the AMS value that can provide an insight into the learning on jumbled joint skeleton data. Hence to accomplish this we downgraded the 1020000 sized jumbled data into 100 feature maps per class per subject in one orientation. Therefore, we have now 100 jumbled feature maps per classs which contains data from a single subject in a particular orientation. Noticeably, we bring uniformity among the experiments in clusters C1 and C2. This is important for getting a deeper insight into the performance of JUMBLESKEL when compared to MONOSKEL action recognition. Finally, we performed the experiment on all subjects in all orientations and the results were averaged across each experiment. Similar to the previous section, the training samples are incremental with a positive rate of 20 per experiment. The remaining are used for testing. In each training set 20% are kept for validation. Meanwhile, the same networks are trained from scratch with all the hyper parameters discussed in section III. Moreover, the hyper parameters are kept constant across networks.

Table 4 presents the results of our experimentation on different AMS values from KLHA3D102 data. Here the random joint maps are from a single run of the ''The Fisher-Yates Shuffle'' on GTX1070 8GB GPU. Table 4 is having structural similarly with table 2 to help readers understand the difference between the unshuffled or base joint learning and shuffled mode. The mean recognition accuracy increased with as the number of training samples inputted are increased. Subsequently, it became reasonably consistent in the AMS

TABLE 5. mean recognition accuracies on jumbled features from multiple sources.

Computed mean recognition accuracies on jumbled features from multiple sources							
Datasets	CNN8 [22]	VGG16 [61]	VGG19 [62]	ResNet-50 [63]	GoogLeNet [64]	SENet-154 [65]	SgCNN (Proposed)
HDM05 [12]	78.66	80.26	80.73	81.58	83.40	86.11	88.31
CMU [13]	75.74	77.13	78.17	79.45	81.30	83.82	86.06
MSR Action3D [15]	79.32	83.19	82.31	83.28	85.95	87.52	90.15
NTU RGB D [14]	83.43	85.37	86.06	86.67	88.36	90.97	93.06
KLYOGA3D [17]	84.54	87.13	88.12	88.75	91.69	94.56	96.43
KLHA3D102 [16]	85.98	87.17	88.29	88.83	91.47	93.70	95.49

TABLE 6. Performance evaluation of QJVM and SgCNN on KLHA3D-102 from multiple sources.

Evaluating QJVM and gCNN on KLHA3D-102							
Machines Used for Experiment	CNN8 [22]	VGG16 [61]	VGG19 [62]	ResNet-50 [63]	GoogLeNet [64]	SENet-154 [65]	SgCNN (Proposed)
4GB Nvidia 940MX Windows	78.24	79.68	80.57	83.64	85.47	85.25	86.45
4GB Nvidia 940MX Linux	79.05	80.24	81.06	83.94	85.46	86.54	87.12
8GB Nvidia GTX 1070 Windows	78.19	79.83	80.41	83.46	84.38	84.87	86.56
4GB Nvidia Quadro P100 Windows	77.96	79.58	79.84	83.24	84.61	84.29	85.42
6GB Nvidia Tesla k20 Linux	78.42	80.07	80.98	83.57	85.07	85.45	86.94

range of 80 and above. This happened for only SgCNN where as the other networks it was beyond 80 samples. Here we have to indeed forced to increase the number of jumbled joint features to 90 for training the other networks. GoogleNet and VGG 16 has achieved in 80, whereas others reached a maximum accuracy at 88 jumbled joint features per class.

Further, the SgCNN was able to achieve this results with comparatively less computational costs over the other networks. Additionally, there was no vanishing gradients problem in our network which was encountered by us when training the state-of-the-art models and have to eventually retrain them by applying weight regularizers. The reason for better performance in SgCNN has been attributed to the multiple hierarchical filter Kernels applied across convolutional layers. In short, variations in joints of the skeleton during experimentation can effectively be learned by a deep network which can then identify an differently ordered joint action class with around 90% accuracy.

2) JUMBLESKEL PERFORMANCE ACROSS DATASETS

Table 5 shows the computed mean recognition accuracies on jumbled features from multiple sources. The variations in results were found to be similar to that of table 3. However, the accuracies across features and networks have been approximately equal to that of that of table 4. This consistency in mean accuracy can be quantified to the fact that the networks have learned to characterize the jumbled features and they have now become powerful enough to generalize on the noisy skeletal samples. Notably, table 5 has better performing networks compared to table 3. This is indeed an interesting observation showing that jumble joint representations are helpful in improving the quality of skeletal action recognition systems. The final cluster C3 evaluates the universality of the proposed framework for action recognition.

D. C3: OMNIPRESENCE OF JUMBLESKEL FRAMEWORK

This part of the work evaluates the proposed concept of joint independent learning in deep networks for skeletal action recognition through iterative execution on multiple machines. For this purpose we used 5 types of GPUs located on 5 different machines, i.e. 3 laptops, workstation and a high performance computing (HPC) center located at our university campus. The machines used are, NVIDIA GTX1070 8GB GPU with a 16GB RAM, 4GB Quadro P100, 6GB TESLA K20M and two 940MX 4GB GPUs. All these GPUs are present on 5 different machines with different memory configurations. However, all are from NVIDIA and data management is performed using CUDA architecture.

The entire framework is executed from end-to-end on each of these machines. The feature maps were extracted on each machine and they are used for training from scratch. This operation was necessary to ensure the proposed joint order independent learning is actually implementable in real sense. Since, the skeletal joint shuffling process is random, which generates different joint orders during each routine execution either on the same machine or on different machine. This entire cluster is using only KLHA3D102 dataset as input. All the hyper parameters were made constant across machines

and iterations. Each model is executed 5 times on 5 different machines from feature extraction to action prediction.

The mean recognition accuracy has been averaged across the datasets and the iterations. Therefore, we present table 6 with mean accuracies on different GPUs for QJVM feature maps trained on multiple models. Since we have the AMS for our KLHA3D102 dataset, we used 80 training samples for training each of these networks and tested with the remaining 20 samples of JUMBLESKEL. The resulting accuracies are averaged across both datasets and iterations on each machine. We found that the recognition accuracies have fractional deviation in each iteration on a machine and hence it was averaged across iterations along with the dataset samples. The results in table 6 are a replica of the table 4 and the essence of constant joint ordering can be replaced by capricious. In short, we found through experimentation and subsequent analysis that deep learning models can be trained on a random joint set feature maps to estimate skeletal actions with any joint representations. However, we estimate that the results will be slightly different on multiple machines as we have demonstrated in this cluster.

In conclusion, the minimum number of randomly ordered joint maps required for achieving independence is found to be above 80 samples per class. This has indeed achieved maximum accuracy of 86.56% on our proposed QJVMs and SgCNN was the highest among the state-of-the-art methods. Further increasing the AMS has improved the accuracies across models and features for all the considered skeletal action datasets fractionally. However, the training and validation losses were staggering around 0.00143 and 0.00054 after 80 training samples. All the models were run for 200 epochs and the accuracies reported in this work are average rates at 200th epoch.

E. THREAT ANALYSIS

Adding different randomly generated joint orders as test samples called as un-controlled group. This un-controlled group is constantly tested against the control group of test samples from the actual datasets. If the results from the un-controlled group are close enough to the controlled test group, the model outputs will be affected. In our experimentation the recognition accuracies from these two groups were separated by a margin of $\pm 2.6\%$ across all datasets. This process has nullified the internal threats to a large extent.

Testing on multiple machines has been performed to counter the external threats imposed by generating the random joint sequences on different machines and testing the resulting action feature maps on multiple machines as described in the last cluster C3.

V. CONCLUSION

A joint order independent learning method for skeletal based action recognition is proposed, evaluated and validated along with recognition method. Skeletal action datasets from mocap and Kenect are used for experimentation and analysis.

Further, the joint order variational data is created using random shuffling mechanism on the base skeletal joint data, called as jumbled joints. The conclusions are threefold: one, the new spatio temporal joint feature maps QJVMs has shown to have discriminating pixel patterns across closely related actions. Subsequently, a Spectrally Graded CNN architecture is developed for image classification tasks by using multiple filter Kernel sizes which enhances the non linearity in the learning through hierarchical receptive fields across the network. Second, the MONOSKEL training and testing with different feature maps from various datasets on deep learning frameworks has shown that the minimum AMS necessary was in the range of 60 to 80 samples per class. The mean accuracies across various skeletal action datasets was found to be in the range of 88 to 97% across features that have some kind of joint to joint relationships and if more joints are involved in a relationship the better are the pixel patterns. Thirdly, this study facilitated the discovery that it is possible to use different joint orderings (JUMBLESKEL) for skeletal action recognition. It further points to a better joint relationships in features greatly increases the networks capacity to generalize better. We also found through exhaustive experimentation on multiple machines that a AMS in the range of 80 and above training samples per class are necessary to develop a omnipresence 3D skeletal action recognition system. This study concludes that it is possible to develop a joint order independent skeletal action recognition system with joint relationship feature maps and deep learning networks.

REFERENCES

[1] W. Li, Z. Zhang, and Z. Liu, "Action recognition based on a bag of 3D points," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Workshops*, Jun. 2010, pp. 9–14.
[2] A. Hernández-Vela, M. A. Bautista, X. Perez-Sala, V. Ponce, X. Baró, O. Pujol, C. Angulo, and S. Escalera, "BoVDW: Bag-of-visual-and-depth-words for gesture recognition," in *Proc. 21st Int. Conf. Pattern Recognit. (ICPR)*, 2012, pp. 449–452.
[3] J. Yan, X.-C. Yin, W. Lin, C. Deng, H. Zha, and X. Yang, "A short survey of recent advances in graph matching," in *Proc. ACM Int. Conf. Multimedia Retr.*, Jun. 2016, pp. 167–174.
[4] D. A. Kumar, A. S. C. S. Sastry, P. V. V. Kishore, and E. K. Kumar, "Indian sign language recognition using graph matching on 3D motion captured signs," *Multimedia Tools Appl.*, vol. 77, no. 24, pp. 32063–32091, Dec. 2018.
[5] M. R. Aghamohammadi and M. R. Salimian, "A three stages decision tree-based intelligent blackout predictor for power systems using brittleness indices," *IEEE Trans. Smart Grid*, vol. 9, no. 5, pp. 5123–5131, Sep. 2017.
[6] C. S. Pitombo, A. D. de Souza, and A. Lindner, "Comparing decision tree algorithms to estimate intercity trip distribution," *Transp. Res. C, Emerg. Technol.*, vol. 77, pp. 16–32, Apr. 2017.
[7] H. Wang and L. Wang, "Learning robust representations using recurrent neural networks for skeleton based action classification and detection," in *Proc. IEEE Int. Conf. Multimedia Expo Workshops (ICMEW)*, Jul. 2017, pp. 591–596.
[8] N. McLaughlin, J. M. del Rincon, and P. Miller, "Video person re-identification for wide area tracking based on recurrent neural networks," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 9, pp. 2613–2626, Sep. 2019.
[9] K. Greff, R. K. Srivastava, J. Koutnìk, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A search space Odyssey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2222–2232, Oct. 2016.
[10] Q. Ke, S. An, M. Bennamoun, F. Sohel, and F. Boussaid, "SkeletonNet: Mining deep part features for 3-D action recognition," *IEEE Signal Process. Lett.*, vol. 24, no. 6, pp. 731–735, Jun. 2017.
[11] C. Li, Y. Hou, P. Wang, and W. Li, "Joint distance maps based action recognition with convolutional neural networks," *IEEE Signal Process. Lett.*, vol. 24, no. 5, pp. 624–628, May 2017, doi: 10.1109/LSP.2017.2678539.

[12] M. Müller, T. Röder, M. Clausen, B. Eberhardt, B. Krüger, and A. Weber, "Documentation mocap database HDM05," Institut für Informatik II, Universität Bonn, Tech. Rep. 7, 2007.
[13] M. Barnachon, S. Bouakaz, B. Boufama, and E. Guillou, "Ongoing human action recognition with motion capture," *Pattern Recognit.*, vol. 47, no. 1, pp. 238–247, 2014.
[14] 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. (CVPR)*, Jun. 2016, pp. 1010–1019.
[15] O. Oreifej and Z. Liu, "HON4D: Histogram of oriented 4D normals for activity recognition from depth sequences," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 716–723.
[16] P. V. V. Kishore, D. G. Perera, M. T. K. Kumar, D. A. Kumar, and E. K. Kumar, "A quad joint relational feature for 3D skeletal action recognition with circular CNNs," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Oct. 2020, pp. 1–5.
[17] T. K. K. Maddala, P. V. V. Kishore, K. K. Eepuri, and A. K. Dande, "YogaNet: 3-D yoga asana recognition using joint angular displacement maps with ConvNets," *IEEE Trans. Multimedia*, vol. 21, no. 10, pp. 2492–2503, Oct. 2019, doi: 10.1109/tmm.2019.2904880.
[18] P. Kishore, D. A. Kumar, A. C. S. Sastry, and E. K. Kumar, "Motionlets matching with adaptive kernels for 3-D Indian sign language recognition," *IEEE Sensors J.*, vol. 18, no. 8, pp. 3327–3337, Apr. 2018.
[19] F. Han, B. Reily, W. Hoff, and H. Zhang, "Space-time representation of people based on 3D skeletal data: A review," *Comput. Vis. Image Understand.*, vol. 158, pp. 85–105, May 2017.
[20] L. L. Presti and M. L. Cascia, "3D skeleton-based human action classification: A survey," *Pattern Recognit.*, vol. 53, pp. 130–147, May 2016.
[21] H.-B. Zhang, Y.-X. Zhang, B. Zhong, Q. Lei, L. Yang, J.-X. Du, and D.-S. Chen, "A comprehensive survey of vision-based human action recognition methods," *Sensors*, vol. 19, no. 5, p. 1005, 2019.
[22] E. K. Kumar, P. V. V. Kishore, A. S. C. S. Sastry, M. T. K. Kumar, and D. A. Kumar, "Training CNNs for 3-D sign language recognition with color texture coded joint angular displacement maps," *IEEE Signal Process. Lett.*, vol. 25, no. 5, pp. 645–649, May 2018.
[23] M. Draelos, Q. Qiu, A. Bronstein, and G. Sapiro, "Intel realsense=Real low cost gaze," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2015, pp. 2520–2524.
[24] V. Silva, F. Soares, J. S. Esteves, J. Figueiredo, C. Santos, and A. P. Pereira, "Happiness and sadness recognition system—Preliminary results with an *Intel RealSense* 3D sensor," in *CONTROLO 2016*. Cham, Switzerland: Springer, 2017, pp. 385–395.
[25] 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.*, Jun. 2012, pp. 1290–1297.
[26] J. Cavazza, A. Zunino, M. S. Biagio, and V. Murino, "Kernelized covariance for action recognition," in *Proc. 23rd Int. Conf. Pattern Recognit. (ICPR)*, Dec. 2016, pp. 408–413.
[27] H. Rahmani, A. Mian, and M. Shah, "Learning a deep model for human action recognition from novel viewpoints," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 3, pp. 667–681, Mar. 2017.
[28] W. Zhu, C. Lan, J. Xing, W. Zeng, Y. Li, L. Shen, and X. Xie, "Co-occurrence feature learning for skeleton based action recognition using regularized deep LSTM networks," in *Proc. 13th AAAI Conf. Artif. Intell.*, 2016, pp. 1–7.
[29] Y. Du, Y. Fu, and L. Wang, "Representation learning of temporal dynamics for skeleton-based action recognition," *IEEE Trans. Image Process.*, vol. 25, no. 7, pp. 3010–3022, Jul. 2016.
[30] N. Akhtar and A. Mian, "Threat of adversarial attacks on deep learning in computer vision: A survey," *IEEE Access*, vol. 6, pp. 14410–14430, 2018.
[31] T.-H. Yang, T.-H. Tseng, and C.-P. Chen, "Recurrent neural network-based language models with variation in net topology, language, and granularity," in *Proc. Int. Conf. Asian Lang. Process. (IALP)*, Nov. 2016, pp. 71–74.
[32] K. S. Tai, R. Socher, and C. D. Manning, "Improved semantic representations from tree-structured long short-term memory networks," 2015, *arXiv:1503.00075*. [Online]. Available: http://arxiv.org/abs/1503.00075
[33] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
[34] M. Liang and X. Hu, "Recurrent convolutional neural network for object recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3367–3375.
[35] T. Song, W. Zheng, P. Song, and Z. Cui, "EEG emotion recognition using dynamical graph convolutional neural networks," *IEEE Trans. Affect. Comput.*, vol. 11, no. 3, pp. 532–541, Jul. 2020.
[36] Z. Cui, R. Ke, Z. Pu, and Y. Wang, "Deep bidirectional and unidirectional LSTM recurrent neural network for network-wide traffic speed prediction," 2018, *arXiv:1801.02143*. [Online]. Available: http://arxiv.org/abs/1801.02143
[37] A. Ullah, J. Ahmad, K. Muhammad, M. Sajjad, and S. W. Baik, "Action recognition in video sequences using deep bi-directional LSTM with CNN features," *IEEE Access*, vol. 6, pp. 1155–1166, 2017.
[38] C. Li, P. Wang, S. Wang, Y. Hou, and W. Li, "Skeleton-based action recognition using LSTM and CNN," in *Proc. IEEE Int. Conf. Multimedia Expo Workshops (ICMEW)*, Jul. 2017, pp. 585–590.
[39] B. Meng, X. J. Liu, and X. Wang, "Human action recognition based on quaternion spatial-temporal convolutional neural network and LSTM in RGB videos," *Multimedia Tools Appl.*, vol. 77, no. 20, pp. 26901–26918, 2018.
[40] Y. Shi, Y. Tian, Y. Wang, and T. Huang, "Sequential deep trajectory descriptor for action recognition with three-stream CNN," *IEEE Trans. Multimedia*, vol. 19, no. 7, pp. 1510–1520, Jul. 2017.
[41] J. Liu, A. Shahroudy, D. Xu, A. K. Chichung, and G. Wang, "Skeleton-based action recognition using spatio-temporal LSTM network with trust gates," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 12, pp. 3007–3021, Dec. 2017.
[42] M. Z. Uddin, N. D. Thang, and T.-S. Kim, "Human activity recognition via 3-D joint angle features and hidden Markov models," in *Proc. IEEE Int. Conf. Image Process.*, Sep. 2010, pp. 713–716.
[43] J. Liu, N. Akhtar, and A. Mian, "Skepxels: Spatio-temporal image representation of human skeleton joints for action recognition," 2017, *arXiv:1711.05941*. [Online]. Available: http://arxiv.org/abs/1711.05941
[44] T. Krosshaug and R. Bahr, "A model-based image-matching technique for three-dimensional reconstruction of human motion from uncalibrated video sequences," *J. Biomech.*, vol. 38, no. 4, pp. 919–929, Apr. 2005.
[45] M. Jiang, J. Kong, G. Bebis, and H. Huo, "Informative joints based human action recognition using skeleton contexts," *Signal Process., Image Commun.*, vol. 33, pp. 29–40, Apr. 2015.
[46] P. Wang, W. Li, C. Li, and Y. Hou, "Action recognition based on joint trajectory maps with convolutional neural networks," *Knowl.-Based Syst.*, vol. 158, pp. 43–53, Oct. 2018.
[47] X. Yang and Y. L. Tian, "EigenJoints-based action recognition using naive-bayes-nearest-neighbor," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Workshops*, Jun. 2012, pp. 14–19.
[48] Y. Du, Y. Fu, and L. Wang, "Skeleton based action recognition with convolutional neural network," in *Proc. 3rd IAPR Asian Conf. Pattern Recognit. (ACPR)*, Nov. 2015, pp. 579–583.
[49] M. Naveenkumar and S. Domnic, "Deep ensemble network using distance maps and body part features for skeleton based action recognition," *Pattern Recognit.*, vol. 100, Apr. 2020, Art. no. 107125.
[50] H. Wang and L. Wang, "Modeling temporal dynamics and spatial configurations of actions using two-stream recurrent neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 499–508.
[51] 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.
[52] E. K. Kumar, P. V. V. Kishore, M. T. K. Kumar, D. A. Kumar, and A. S. C. S. Sastry, "Three-dimensional sign language recognition with angular velocity maps and connived feature resnet," *IEEE Signal Process. Lett.*, vol. 25, no. 12, pp. 1860–1864, Dec. 2018.
[53] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri, "Actions as space-time shapes," in *Proc. 10th Int. Conf. Comput. Vis. (ICCV)*, Oct. 2005, vol. 1, vol. 2, pp. 1395–1402.
[54] R. Christoph and F. A. Pinz, "Spatiotemporal residual networks for video action recognition," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3468–3476.
[55] M. Tayel, G. Dawood, and H. Shawky, "Block cipher S-box modification based on Fisher-yates shuffle and Ikeda map," in *Proc. IEEE 18th Int. Conf. Commun. Technol. (ICCT)*, Oct. 2018, pp. 59–64.
[56] H. Mureşan and M. Oltean, "Fruit recognition from images using deep learning," *Acta Universitatis Sapientiae, Inf.*, vol. 10, no. 1, pp. 26–42, Aug. 2018.
[57] L. Bossard, M. Guillaumin, and L. Van Gool, "Food-101–mining discriminative components with random forests," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2014, pp. 446–461.
[58] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset," Caltech Comput. Vis. Lab., California Inst. Technol., Pasadena, CA, USA, Tech. Rep., 2007.

[59] H. M. Afkham, A. T. Targhi, J.-O. Eklundh, and A. Pronobis, "Joint visual vocabulary for animal classification," in *Proc. 19th Int. Conf. Pattern Recognit.*, Dec. 2008, pp. 1–4.
[60] M. D. Rodriguez, J. Ahmed, and M. Shah, "Action MACH a spatio-temporal maximum average correlation height filter for action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–8.
[61] H. Qassim, A. Verma, and D. Feinzimer, "Compressed residual-VGG16 CNN model for big data places image recognition," in *Proc. IEEE 8th Annu. Comput. Commun. Workshop Conf. (CCWC)*, Jan. 2018, pp. 169–175.
[62] M. Mateen, J. Wen, Nasrullah, S. Song, and Z. Huang, "Fundus image classification using VGG-19 architecture with PCA and SVD," *Symmetry*, vol. 11, no. 1, p. 1, Dec. 2018.
[63] E. Rezende, G. Ruppert, T. Carvalho, F. Ramos, and P. de Geus, "Malicious software classification using transfer learning of ResNet-50 deep neural network," in *Proc. 16th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2017, pp. 1011–1014.
[64] P. Ballester and R. M. Araujo, "On the performance of GoogLeNet and AlexNet applied to sketches," in *Proc. 13th AAAI Conf. Artif. Intell.*, 2016, pp. 1–5.
[65] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7132–7141.

M. TEJA KIRAN KUMAR (Student Member, IEEE) received the B.Tech. degree in electronics and communication engineering from the Vignan's Institute of Information Technology affiliated to JNT University, Kakinada, India, in 2013, and the M.Tech. degree in communication engineering and signal processing from Nagarjuna University, Guntur, India, in 2015. He is currently pursuing the Ph.D. degree with the Koneru Lakshmaiah Education Foundation, India. His research interests include deep learning, object detection, action recognition, and biomechanical analysis.

P. V. V. KISHORE (Senior Member, IEEE) received the master's degree from the Cochin University of Science and Technology and the Ph.D. degree from the Andhra University College of Engineering, in 2013.

He is currently a Professor of image and video processing with the Department of Electronics and Communications Engineering, where he manages the Image, Speech and Signal Processing Research Group. He is also the Chair of the Biomechanics and Vision Computing Research Center. His works focus on machine learning, biomechanics, artificial intelligence, human action analysis, and sign language machine translation. He is an enthusiast in developing new innovations in the areas of computer vision and machine learning areas. He has authored several publications in these fields. His research interest includes how motion capture data models can effectively model low end video objects in real time for better recognition and analysis.

B. T. P. MADHAV (Senior Member, IEEE) was born in Andhra Pradesh, India, in 1981. He received the B.Sc., M.Sc., M.B.A., and M.Tech. degrees from Nagarjuna University, Andhra Pradesh, in 2001, 2003, 2007, and 2009, respectively, and the Ph.D. degree in antennas from K. L. University, in 2015. From 2003 to 2007, he worked as a Lecturer. From 2007 to 2011, he worked as an Assistant Professor. From 2011 to 2015, he worked as an Associate Professor. Since August 2015, he has been working as a Professor in electronics and communication engineering. He has published more than 300 papers in international and national journals and conferences. His research interests include antennas, liquid crystals applications, and wireless communications. He is a Life Member of ISTE, IACSIT, IRACST, IAENG, and UACEE. He is also a fellow of IAEME. He served as a reviewer for three international conferences. He is a reviewer of several international journals, including Elsevier and Taylor and Francis. He is the Editorial Board Member of 15 journals and acting as a Sub-Editor of *International Journal of Speech Technology* (IJST).

D. ANIL KUMAR (Member, IEEE) received the M.Tech. and Ph.D. degrees from the Koneru Lakshmaiah Education Foundation, Guntur, India. He is currently an Assistant Professor at the Department of Electronics and Communication Engineering, PACE Institute of Technology and Sciences (Autonomous), Ongole, India. His work is mainly focused on the development of 3-D processing algorithms for computer vision applications. His research interests include video processing, computer vision, and sign language machine translation.

N. SASI KALA (Member, IEEE) recieved the B.Tech. and M.Tech. degrees in electronics and communication engineering from JNTU, in 2008, with a focus on digital systems computer electronics, and the Ph.D. degree from the Koneru Lakshmaiah Education Foundation (Deemed to be University), in 2020. She is currently working as an Associate Professor with the Department of Electronics and Communication Engineering, Kamala Institute of Technology and Science, Singapur. She has to her credit a total 22 papers published in reputed national and international journals and conferences. Her research interests include digital image processing, computer vision, and satellite communications. She is a Life Member of IETE, ISTE, IAOIP, and IAENG.

K. PRAVEEN KUMAR RAO (Member, IEEE) recieved the B.E. degree in computer technology from Nagpur University, the M.Tech. degree in computer science engineering (CSE) from Kakatiya University, in 2006, with a focus on software engineering, and the Ph.D. degree from the Vel Tech Rangarajan Dr Sagunthala Research and Development Institute of Science and Technology, in 2020. He is currently working as an Associate Professor with the Department of Computer Science and Engineering, Kamala Institute of Technology and Science, Singapur. He has to his credit a total 30 papers published in reputed national and international journals and conferences. His research interests include wireless networks, mobile ad hoc networks, and distributed networks. He is a Life Member of IETE and ISTE and a member of CSI.

B. PRASAD received the B.Tech. and M.Tech. degrees from JNTUK, India, in 2014, and the Ph.D. degree from the Department of Electronics and Communications Engineering, Andhra University. He is currently a Professor and the HOD of IT and MCA departments from the Vignan's Institute of Information Technology. He is having 20 years of research and teaching experience. His research interests include video processing, computer architecture, and programming languages.