Bayesian Graph Convolution LSTM for Skeleton Based Action Recognition

Rui Zhao[1], Kang Wang[1], Hui Su[1,2], Qiang Ji[1]
[1]RPI, [2]IBM Research
{zhaorui.zju,kangwang.kw}@gmail.com, huisuibmres@us.ibm.com, qji@ecse.rpi.edu

Abstract

We propose a framework for recognizing human actions from skeleton data by modeling the underlying dynamic process that generates the motion pattern. We capture three major factors that contribute to the complexity of the motion pattern including spatial dependencies among body joints, temporal dependencies of body poses, and variation among subjects in action execution. We utilize graph convolution to extract structure-aware feature representation from pose data by exploiting the skeleton anatomy. Long short-term memory (LSTM) network is then used to capture the temporal dynamics of the data. Finally, the whole model is extended under the Bayesian framework to a probabilistic model in order to better capture the stochasticity and variation in the data. An adversarial prior is developed to regularize the model parameters to improve the generalization of the model. A Bayesian inference problem is formulated to solve the classification task. We demonstrate the benefit of this framework in several benchmark datasets with recognition under various generalization conditions.

1. Introduction

In recent years, skeleton based action recognition attracts an increasing attention as the quality of 2D/3D pose estimation methods improve [56, 48, 8]. Derived from RGB or depth videos, skeleton data provides a succinct and informative representation for the action execution. Pure skeleton based action recognition can achieve quite good results [17, 57] despite being a much lower dimensional representation. These results make skeleton an attractive modality for action recognition. However, recognizing actions from skeleton data remains challenging partially due to the complex nature of human motion dynamics.

In particular, there are three major factors that contribute to the complex dynamics. First, human motion often involves coordination of different body parts, which introduces spatio-temporal dependencies among body parts. For example, to complete a bowling action, the four limbs need to move in a particular way to complete the action of throwing the ball, while keeping the balance of the whole body. Furthermore, the kinematics of a motion pattern can be nonlinear and complex, *e.g.*, boxing and dancing. Second, there exists long-term dependency in motion pattern. The body pose of different actions may be similar at certain time. However, the cause of such pose and its specific meaning in the context depend on the motion in the past. Finally, there exists significant subject-dependent variation due to different habits. Different subjects may perform the same action differently in terms of the extent and speed of the movement. In a word, the coupling of spatial, temporal and subject factors increases the challenge of recognizing actions from skeleton data.

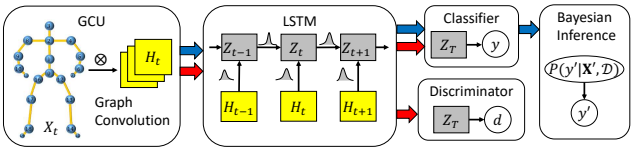

Figure 1. Overview of our framework. The red and blue arrows show the flow of training and testing data, respectively. Details of each component are discussed in Section 3.

In this paper, we propose an end-to-end trainable framework that combines neural networks with probabilistic modeling to simultaneously handle the three factors contributing to the complex dynamics. More specifically, we propose a Bayesian neural network (BNN) model. Our model is first built on a combination of graph convolution with long short-term memory (LSTM) network, in which graph convolution is used to capture the spatial dependency among different body joints and LSTM is used to capture temporal dependency of pose change over time. The neural network model is further extended to a probabilistic model following the Bayesian modeling framework by treating model parameters as random variables. This extension allows the model to better handle randomness in the motion data. To further increase the capability of generalization to unseen data, we introduce an adversarial prior, inspired by adversarial learning to regularize the model parameters. We formulate the classification task as a Bayesian inference problem in order to fully exploit the proposed probabilistic

model, which allows us to reduce overfitting. Our specific contributions are summarized as follows.

- Propose a BNN model that combines graph convolution and LSTM to model complex dynamics in the skeleton data.
- Introduce an adversarial prior to regularize the model parameters.
- Develop a Bayesian inference framework that exploits the distribution of parameters in order to improve robustness and generalization.

2. Related Work

Skeleton based action recognition: There is abundant literature on recognizing human actions from skeleton data [52, 75]. Conventional approaches either focus on developing hand-crafted features [66, 51, 35, 16, 63] or classification models [26, 49, 44, 76, 71, 24, 77]. More recently, neural networks based approaches become the dominate framework due to end-to-end learning of feature representation and classifier [17, 78, 46, 58, 64, 65, 60, 70, 73]. In particular, graph convolution based approaches become popular. Graph convolution extends convolution to arbitrary graph structured data such as skeleton, which has been modeled by graph in prior work [17, 68]. Existing efforts on graph convolution can be divided into two categories [6]. The first type of approach operates in the spatial domain [54, 18, 50]. For each node, a subgraph is constructed based on its connections with other nodes. Then convolution is performed by aggregating the values of each node in the subgraph. The challenge of this approach is the proper choice of ordering of nodes and the handling of cardinality difference among the subgraphs. The second type of approach operates in the spectral domain [31, 15, 39]. This approach is built on the spectral graph theory [30], which provides a spectral domain representation of the graph data. We adopt the spectral-based approach for its flexibility in constructing convolution filters that allow parameter sharing. Furthermore, the construction allows easy extension to deep neural networks. In both types of approaches, the graph structure is often assumed fixed. Several recent work [42, 55] proposed to learn the graph structure from data in order to capture implicit dependencies among joints. Our approach is orthogonal to this effort by leveraging Bayesian framework, which uses a distribution of parameters for inference to better capture the variation in skeleton dynamics.

Modeling complex dynamics: Previous work in modeling complex dynamics can be divided into two major categories, namely probabilistic graphical models (PGM) based approaches and recurrent neural networks (RNN) based approaches. Recently, there is an increasing interest in combining PGM with RNN models to obtain the benefits of both frameworks, namely modeling of stochasticity and automatic feature representation learning. Existing work in this direction can be divided into three main categories. The first category focuses on combining two separate models. For example, CNN or MLP is used to perform automatic feature extraction and PGM such as hidden Markov model (HMM) is used to model dynamics [3, 72, 38]. Another example is using RNN to model dynamics and using PGM to capture structured dependencies in data such as [4, 36, 41, 69]. The second major category focuses on extending RNN and its variants into a probabilistic model. One strategy is to add additional stochastic nodes to RNN to allow the modeling of randomness in dynamic data [2, 29, 13, 20]. Another strategy is extending RNN following the Bayesian framework [21, 22, 19]. The third major category directly converts PGM into an end-to-end trainable model by parameterizing the conditional distribution using NN such as in [40, 14, 9]. Our work focuses on extending RNN based model to a probabilistic model by leveraging Bayesian modeling framework, which treats the parameters of model as random variables with designated prior distribution. In order to capture the interdependencies among body joints, we utilize graph convolution for its capability of simultaneously capturing structured dependency and automatic representation learning. We formulate the classification as a Bayesian inference problem, in which we take both data and model uncertainty into consideration. This allows the model to better generalize to unseen data.

Domain adaptation: Our work is related to generalizing model to different domains such as different groups of subjects. Various approaches have been proposed to reduce the domain shift, for example, by minimizing the maximum mean discrepancy [62] or the correlation distance [59]. In [23, 5, 61], the authors adopted the idea of adversarial learning by introducing additional domain classifier to differentiate samples from the source to the target domain. The idea is to promote features that are domain-invariant such that data from different domains are indistinguishable by the domain classifier. However, these approaches are mainly designed for non-sequential data. In this work, we incorporate adversarial learning into our Bayesian inference framework and formulate it as a prior, which helps regularizing the parameter distribution. Overall, the proposed Bayesian graph convolution LSTM can capture complex dynamics as well as generalizing across subjects and datasets.

3. Methods

We first describe the details of applying graph convolution to skeleton data to extract a structure-aware representation. Then we describe the proposed Bayesian GC-LSTM model and the adversarial prior, followed by the Bayesian inference formulation.

3.1. Representation Learning from Skeletons

Skeleton graph is defined by a set of nodes and a set of edges that connect different nodes. The nodes correspond to different body joints. The edges represent the connection among nodes and they are often determined based on anatomy structure. For example, the elbow joint is connected with the shoulder joint. Artificial connections can also be introduced. For example, Liu *et al.* [46] constructed the edges based on a tree traversal order of the skeleton. Tang *et al.* [60] added additional edges among limbs such as hands and feet, whose motions are highly correlated. In this work, we follow the common design of graph based on anatomy. Specifically, we define an undirected graph at each time step $\mathbf{G}_t = \{\mathbf{X}_t, \mathbf{E}_t\}$. $\mathbf{X}_t = \{X_{t1}, X_{t2}, ..., X_{tN}\}$ is the set of nodes at time t, where each node represents a body joint and N is the total number of joints. $\mathbf{E}_t = \{(X_{ti}, X_{tj}) : X_{ti}, X_{tj} \in \mathbf{X}_t, X_{ti} \sim X_{tj}\}$ is the set of edges in the graph, where $X_{ti} \sim X_{tj}$ means the node i and node j are connected with an undirected edge. An example of the skeleton graph is depicted in Figure 2. $\mathbf{E}_t$ can be specified by the adjacency matrix $A_t \in \mathbb{R}^{N\times N}$.

$$A_t(i,j) = \begin{cases} 1, & \text{if } (X_{ti}, X_{tj}) \in \mathbf{E}_t \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

For each node X_{ti}, the associated observations are 3D joint position and speed. The speed is obtained by computing the change of position between consecutive time steps. Therefore, each node has a 6-dimensional observations and $\mathbf{X}_t \in \mathbb{R}^{N\times 6}$. Similar representation is also used in [67, 70]. Position and speed are complementary to each other in representing the motion since they capture the first and the second order kinematics of the motion, respectively. The graph provides a concise way to specify the dependency among different joints. We assume the graph structure does not change over time, *i.e.*, $\mathbf{G}_t$ remains the same for all t.

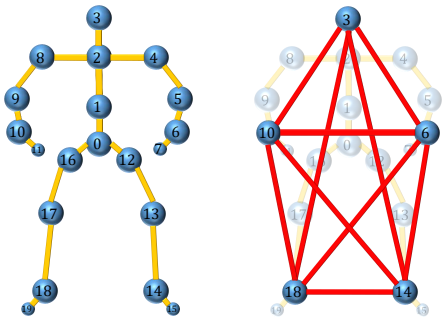

Figure 2. Skeleton graph, consisting of body joints and edges connecting two joints. Left: *local graph*. Right: *global graph*.

Graph convolution defines a convolution operation on data that are specified with an arbitrary graph structure. A special case of graph convolution is applied to images, which can be viewed as a graph with grid structure. Performing convolution on images is straightforward due to the regularity of grid, which provides a homogeneous connection structure for each node. The invariance of structure allows parameter-sharing where the same convolution kernel can be used for different nodes.

One way to perform general graph convolution is derived from spectral graph theory [30], which allows the graph convolution to be carried out in the spectral domain. The key operation is computing the graph Fourier transform, which is equivalent to computing an expansion of the graph with respect to the eigenfunctions of the graph Laplacian. Defferrard *et al.* [15] proposed to use an efficient approximation to compute the graph Fourier transform based on the K^{th}-order Chebyshev polynomial. Kipf and Welling [39] further reduces K to 1, resulting a linear approximation of graph convolution, where the convolution only depends on the neighboring nodes that are directly connected to the target node. The operation of one graph convolutional layer is defined as follows.

$$H^{(l+1)} = \sigma(\sum_{k=1}^{K} F_k H^{(l)} W_k^{(l)} + b_k^{(l)}) \tag{2}$$

where $H^{(l)} \in \mathbb{R}^{N\times d_l}$ and $H^{(l+1)} \in \mathbb{R}^{N\times d_{l+1}}$ are the input and output of l^{th} layer of graph convolution with feature dimension d_l and d_{l+1}, respectively. $F_k = \tilde{D}_k^{-\frac{1}{2}} \tilde{A}_k \tilde{D}_k^{-\frac{1}{2}} \in \mathbb{R}^{N\times N}$ is the graph kernel, where $\tilde{A}_k = A_k + I$ is the augmented adjacency matrix that contains additional self-connection for each node. $\tilde{D}_k$ is the diagonal degree matrix of $\tilde{A}_k$ with $\tilde{D}_k(i,i) = \sum_j \tilde{A}_k(i,j)$. $W_k^{(l)} \in \mathbb{R}^{d_l \times d_{l+1}}$, $b_k^{(l)} \in \mathbb{R}^{d_{l+1}}$ are the kernel weights and bias. K is the total number of graph kernels. The use of multiple graph kernels allows the model to capture different dependency structures. Two examples of graph are shown in Figure 2. σ is a nonlinear activation function.

Notice that F_k is a symmetric matrix, Eq. (2) implicitly assumes a symmetric contribution within each graph kernel on the convolution results. To further increase the effectiveness of the convolution, we multiply F_k by a learnable mask $M_k \in \mathbb{R}^{N\times N}$, resulting in the following operation.

$$H^{(l+1)} = \sigma(\sum_{k=1}^{K} (M_k \odot F_k) H^{(l)} W_k^{(l)} + b_k^{(l)}) \tag{3}$$

By stacking multiple graph convolutional layers with $H^{(0)} = \mathbf{X}_t$, we can extract useful representation which encodes the dependency specified by the graph.

3.2. Bayesian GC-LSTM

GC-LSTM: We first briefly describe LSTM [32], which is a variant of RNN with a special design on the recurrent

state computation. LSTM helps overcome the insufficiency of RNN in learning long-term dependencies. Suppose we have a sequence of observations $X_t \in \mathbb{R}^D, t = 1, ..., T$, and D is the dimension of each X_t. The corresponding hidden states of LSTM are $Z_t \in \mathbb{R}^Q, t = 1, ..., T$, and Q is the dimension of each Z_t. At each time t, an LSTM cell is used to compute Z_t given X_t and Z_{t-1}. The parameterization of an LSTM cell is referred to [28]. Intuitively speaking, LSTM cell introduces additional nodes with learnable parameters to retain the memory from the past and combine the present input to update the hidden state. We use the output of the last graph convolutional layer at each time t as the input of LSTM, resulting a dynamic model as shown in Figure 3. We call this model GC-LSTM.

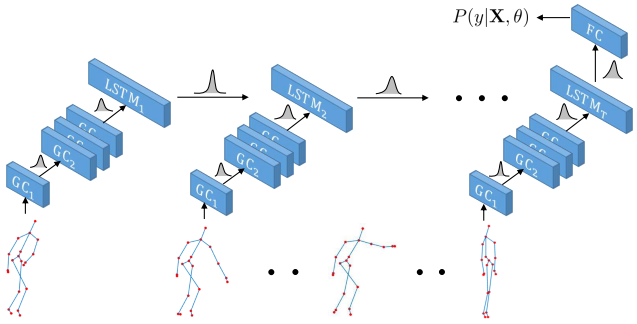

Figure 3. GC-LSTM model architecture.

Bayesian extension: We propose to extend the proposed GC-LSTM into a probabilistic model following Bayesian framework by treating the parameters of GC-LSTM as random variables. Such extension allows the model to better capture the randomness and variation in dynamic data. It has been shown in the literatures that adding stochastic modeling in RNN is beneficial in modeling variability in structured data [2, 13]. Furthermore, the use of Bayesian inference allows us to improve the generalization ability of the model. Bayesian RNN has been proposed in [22, 19] for NLP and image captioning tasks. Our extension also includes a Bayesian treatment of graph convolutional networks, which allows us to further capture the structural dependencies in sequential data.

Suppose $\mathbf{X}$ and y are random variables representing the observed motion sequence and the corresponding class label, respectively. Bayesian GC-LSTM model defines a conditional likelihood $P(y|\mathbf{X}, \theta)$, which specifies the probability of $\mathbf{X}$ being class y. θ includes all the parameters of graph convolutional networks and LSTM. In practice, $P(y|\mathbf{X}, \theta)$ is obtained by feeding the output of the last LSTM cell into a fully connected layer with softmax activation function. Different from non-Bayesian approach, θ is treated as a random variable whose prior distribution $P(\theta|\alpha)$ is specified by hyperparameter α. We treat α as a fixed number following the empirical Bayesian approach [25]. We use standard Gaussian as prior, *i.e.*, $P(\theta|\alpha) = \mathcal{N}(0, I)$.

Classification using Bayesian GC-LSTM can be formulated as a Bayesian inference problem. Given a set of training data $\mathcal{D} = \{\mathbf{X}_i, y_i\}$ and a query of testing data $\mathbf{X}'$, the objective of Bayesian inference is to compute the conditional posterior distribution of target variable y' as follows.

$$P(y'|\mathbf{X}', \mathcal{D}, \alpha) = \int_\theta P(y'|\mathbf{X}', \theta) P(\theta|\mathcal{D}, \alpha) d\theta \approx \frac{1}{M} \sum_{m=1}^{M} P(y'|\mathbf{X}', \theta_m), \theta_m \sim P(\theta|\mathcal{D}, \alpha) \quad (4)$$

Eq. (4) uses Monte Carlo estimation to approximate the integration over θ, which is computationally intractable due to complex model specification. Then the classification criterion is as follows.

$$y^* = \arg\max_{y'} \frac{1}{M} \sum_{m=1}^{M} P(y'|\mathbf{X}', \theta_m) \quad (5)$$

where M is the total number of samples of parameters. Details of performing Bayesian inference is discussed in Section 3.3.

Adversarial prior: The prior distribution $P(\theta|\alpha)$ used in Bayesian extension can be viewed as a regularization that prevents the model parameters to overfit to training data. In particular, the model may overfit to variation caused by subjects. However, the choice of hyperparameter α is determined by heuristic. Inspired by adversarial learning [27], we develop an additional prior to further regularize the model in order to improve the generalization across different subjects. The intuition is that we prefer a feature representation extracted by GC-LSTM to be invariant of subject so that the model will not overfit to the nuisance caused by subject-dependent variation.

Specifically, we introduce a discriminator ϕ that can differentiate whether the feature produced by GC-LSTM share the same subset of subjects. We implement the discriminator as another fully connected layer whose input is the same as the classification layer of GC-LSTM. Let $\{\mathbf{X}^+\} \subset \mathcal{D}$ be a training mini-batch with label $\{y^+\}$ and $\{\mathbf{X}^-\} \subset \mathcal{D}$ be a validation mini-batch with subjects different from $\{\mathbf{X}^+\}$. We have the following posterior distribution for θ.

$$\log P(\theta|\mathcal{D}, \phi, \alpha_\theta) = \log P(y^+|\mathbf{X}^+, \theta) + \log P(\theta|\alpha_\theta) + \log P_D(G(\mathbf{X}^-; \theta)|\phi) + C \quad (6)$$

where $G(\mathbf{X}; \theta)$ represents the output of GC-LSTM before classification layer. $P_D(\cdot|\phi) \in [0, 1]$ represents the probability of belonging to training subjects. C is an unknown normalization constant for the posterior distribution. We see in Eq. (6) that ϕ adds additional regularization to the

posterior of θ in the sense that θ not only needs to improve the classification on labeled training mini-batch but also needs to fool ϕ to produce high likelihood on validation mini-batch. For ϕ, we have the following posterior, where a higher likelihood indicates better differentiation between $\{\mathbf{X}^+\}$ and $\{\mathbf{X}^-\}$.

$$\log P(\phi|\mathcal{D},\theta,\alpha_\phi) = \log P_D(G(\mathbf{X}^+;\theta)|\phi) + \log P(\phi|\alpha_\phi) + \log(1 - P_D(G(\mathbf{X}^-;\theta)|\phi)) + C' \quad (7)$$

where C' is an unknown normalization constant. Comparing Eq. (6) and Eq. (7), we see θ and ϕ are competing against each other. The overall framework is illustrated in Figure 1.

3.3. Bayesian Inference

The key to perform Bayesian inference is generating samples of parameters $\tilde{\theta}$ from the posterior distribution $P(\tilde{\theta}|\mathcal{D},\tilde{\alpha})$. Here $\tilde{\theta} = [\theta, \phi]$ and $\tilde{\alpha} = [\alpha_\theta, \alpha_\phi]$. For Bayesian GC-LSTM, the likelihood is a complex nonlinear function of $\tilde{\theta}$, resulting in intractable posterior of $\tilde{\theta}$. Therefore, approximate inference method such as MCMC and variational method is needed. In this work we use stochastic gradient Hamiltonian Monte Carlo (SGHMC) [11]. HMC can explore the parameter space more efficiently compared to Metropolis-Hastings and Gibbs sampling due to the use of gradient information. Furthermore, the stochastic extension proposed in [11] maintains the convergence property of HMC while allowing the method to scale up to large datasets. One step of SGHMC update is defined as follows.

$$\tilde{\theta}^{(t+1)} = \tilde{\theta}^{(t)} + v^{(t+1)} \quad (8)$$
$$v^{(t+1)} = (1-r)v^{(t)} + \eta\nabla L(\tilde{\theta}) + \epsilon,\ \epsilon \sim \mathcal{N}(0, 2r\eta I)$$

where $r \in (0,1)$ is the momentum coefficient and $L(\tilde{\theta})$ is the unnormalized log posterior defined in Eq. (6) and Eq. (7). Therefore, Eq. (8) essentially converts sampling into an optimization process of performing momentum-based stochastic gradient ascent, which allows us to use existing implementation frameworks of deep neural networks. We use SGHMC instead of variational method in our experiment as it approximates the exact posterior distribution and better computational efficiency. The overall Bayesian inference algorithm is summarized in Algorithm 1. Notice that the same set of samples of parameters is used for inference of all testing data.

Algorithm 1 Bayesian inference of GC-LSTM

Input: $\mathcal{D} = \{\mathbf{X}, y\}$: training data. $\mathcal{D}' = \{\mathbf{X}'\}$: testing data. r: momentum coefficient. T_b: burn-in iterations. T_g: gap iterations between two samples. η: initial learning rate.
Output: $\{y'\}$: predicted labels

```
Initialization: θ̃^(0) ~ N(0, I), v^(0) = t = m = 0
repeat
    Select a mini-batch of {X^+, y^+, X^-} from D
    Update θ̃^(t) using Eq. (8) with specified r and η
    t ← t + 1
    if t ≥ T_b and mod(t, T_g) == 0 then
        θ_m = θ^(t) // Collect as a sample of θ
        m ← m + 1
    end if
until collect enough samples
for Each X'_j ∈ D' do
    Compute P(y'|X'_j, θ_m), ∀m using GC-LSTM
    Solve for y'_j using Eq. (5)
end for
return {y'}
```

4. Experiments

We discuss the experimental evaluation in this section, starting with a description of the datasets and preprocessing. After discussing the implementation, we perform an ablation study and a generalization experiment. Finally, we compare with state-of-the-art on selected benchmark datasets.

4.1. Datasets

MSR Action3D [43] is one of the earliest multi-modal action recognition dataset. There are 20 actions performed by 10 subjects with a total number of 557 sequences. **UTD MHAD** [10] is another multi-modal human action dataset collected via Kinect and wearable sensors. There is a total number of 861 sequences, which contains 27 actions performed by 8 subjects. **SYSU** [34] is a human activity dataset collected from 40 different subjects on 12 activities, which involve manipulating objects such as phone and chair. There are 480 videos in total. **NTU RGB-D** [53] is currently one of the largest human action/activity datasets. It contains 60 categories, which are collected from 40 different subjects. The total number of sequences is 56,880. We only use the skeleton data in each dataset for all the experiments.

Pre-processing: We normalize the data by subtracting the torso joint position from each joint so that the skeleton is translation invariant. We further resize the bone length to a fixed reference length while maintaining the same joint angle so that the skeleton is scale invariant. The same processing is applied for each dataset and used by different variants of our model. In case two people involved in one action, the skeleton with active motion is used.

4.2. Implementation

We implement the model using Tensorflow [1]. A graph convolutional unit (GCU) is implemented by matrix multiplication as defined in Eq. (3). The convolution is followed by a batch normalization and an activation function, which we use ReLU. We stack multiple GCUs together where the output of lower level is the input of the next higher level. In

experiment, we found 4 GCUs perform the best. We also add a skip connection between every two GCUs. At each time step t, the raw skeleton representation $\mathbf{X}_t$ is fed into the same shared GCU stack. The number of output channels for each GCU is 64 for NTU and 8 for other datasets. The output of the last GCU is then flattened and fed into the LSTM cell of current time, which unrolls to form a dynamic model. We use state size of 256 for NTU and 128 for all other datasets. Dropout is applied to the input and output of LSTM to further prevent overfitting. The output of the last time step of LSTM is used as the final representation of the motion sequence. We feed it into a fully connected layer with softmax activation function to generate class-wise probability vector, *i.e.*, $P(y|\mathbf{X}, \theta)$. The same representation is fed into a single unit with sigmoid activation function to generate the subject discriminative probability, *i.e.*, $P_D(\cdot|\phi)$. We use momentum-based gradient ascent. The initial learning rate is 0.001. We decay the rate by 0.9 every 100 epochs. A batch size of 256 is used for NTU and 32 for all other datasets with an equal split of samples of training and validation in a mini-batch.

4.3. Ablation Study

We perform an ablation study using UTD dataset. We evaluate how different components of the model affect the final performance.

First, we evaluate how graph convolution contributes to the final performance. We consider two baseline approaches. The first one (*no graph*) directly feeds all the joint data at each time step to LSTM input gate. The second one (*mean-field graph*) defines a graph with an empty edge set, *i.e.*, all the nodes in the graph are isolated without any connections. In such case, graph convolution reduces to a 1×1 convolution that operates along different channels of each node. For our proposed approach, we evaluate three variants. The first one (*local graph*) defines the graph based on skeleton anatomy. The second one (*global graph*) only adds edges between limbs and head, while treating all other joints as independent. Finally, *joint graph* combines previous two graphs by computing the sum of the output of both local and global graphs, *i.e.*, two graph kernels. The two graph structures are shown in Figure 2.

In this set of experiments, we only vary the configuration of graph convolution, while fixing other part of the model such as LSTM state size and optimization scheme. We repeat experiment under each setting five times and report the average to account for random initializations. The results are shown in Table 1.

The results have following indications. First, we see that *local graph* outperforms the variants without graph convolution. This indicates the effectiveness of capturing the dependencies among different joints. *Global graph* by itself does not perform well as it ignores the local dependencies. Combining the two graphs achieves the best performance indicates that *local graph* and *global graph* contains complementary information to each other. Second, *no graph* setting achieves better performance compared to simple *mean-field graph*. This indicates that a proper construction of the graph is important and learning a representation based on independent structure should be avoided.

Table 1. Effect of graph convolution.

Configuration	# of edges	Accuracy
No graph	N/A	85.0
Mean-field	0	82.2
Local graph	19	87.2
Global graph	10	81.5
Joint graph	29	**92.1**

For the next set of experiments, we evaluate different choices of dynamic modeling architectures including vanilla RNN, GRU [12], LSTM, bi-directional LSTM and stacked LSTM. We use *joint graph* for convolution. Among all the variants, vanilla RNN has the worst performance, which demonstrates the necessity of modeling long-term temporal dynamics. GRU is close to LSTM. Different variations of LSTM achieves comparable performance with the best configuration being a single LSTM layer. We use one-layer LSTM for the remaining experiments and leave a thorough exploration of LSTM architectures as future work.

Table 2. Effect of RNN architecture.

Architecture	Accuracy
Vanilla RNN	66.4
GRU	89.7
LSTM	**92.1**
2-stacked LSTM	92.0
Bi-directional LSTM	90.8

For the next set of experiments, we evaluate the effect of Bayesian inference by comparing against point estimation approach. *Joint graph* and single-layer LSTM are used to construct the model. For point estimation approach, we consider two different methods, namely maximum likelihood (ML) and maximum a posterior (MAP). For Bayesian inference approach, we consider the case with and without using adversarial prior (AP) during posterior sampling. To further demonstrate the benefit of using Bayesian inference, we create additional testing data which add perturbation to the original testing data. We consider two types of perturbation. The first one is applying random rotation to the original testing data to mimic the variation in camera view. The second one is adding Gaussian random noise to the joint position to mimic poor estimation of pose. The results are shown in Table 3.

From the results we see that Bayesian inference method

Table 3. Effect of Bayesian inference under rotation (R) and additive noise (N).

Perturbation	Clean	Only R	Only N	R + N
ML	86.2	62.8	77.7	65.1
MAP	85.2	78.1	86.1	77.9
Bayesian	87.4	78.8	86.9	82.8
Bayesian + AP	**92.1**	**86.1**	**87.5**	**85.9**

outperforms both point estimation methods. We also observe that MAP outperforms ML in perturbed testing data, which indicates that it is beneficial to place a prior distribution on the parameters to improve the robustness. Bayesian inference with adversarial prior achieves the best results among all variants, which demonstrates the effectiveness of adversarial prior in helping the model to adapt to new data. Comparing the results between clean data and perturbed data, we observe that Bayesian inference approaches yield smaller decrease in performance. This shows the benefit of using different sets of parameters increase the robustness of the model.

To evaluate the number of samples needed for Bayesian inference, we plot the accuracy versus the number of samples under clean testing data case and the result is shown in Figure 4. We observe that the performance becomes saturate after about 50 samples. We use 100 samples for the remaining experiments.

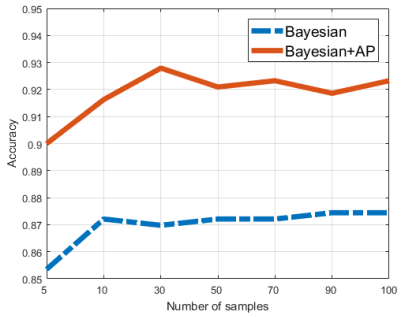

Figure 4. Bayesian inference results under different number of samples of parameters.

4.4. Generalization

We demonstrate the capability of the proposed framework in generalization by performing a cross-dataset experiment, where the training and testing data come from different datasets. Therefore, there exist significant variations between training and testing data. In the first experiment we use MSR and UTD datasets, which share 10 actions in common[1]. Figure 5 shows examples of the same actions from two different datasets. We observe not only substantial variations in the action executed by different subjects but also different subject postures caused by different data collection settings. Our baseline is Bayesian inference without adversarial prior. We also compare with three state-of-the-art methods. R3DG [63] uses sophisticated hand-crafted features and SVM classifier. DLSTM [78] is a deep learning based approach. [73] is a graph convolution based approach. The results are shown in Table 4. We observe substantial improvement by using AP in our method, which demonstrates the effectiveness of the adaptation.

Table 4. Generalization across different datasets.

Train	MSR	UTD	NTU		Avg.
Test	UTD	MSR	UTD	MSR	
R3DG [63]	66.5	59.9	69.8	64.5	65.2
DLSTM [78]	66.8	50.0	61.0	62.6	60.1
STGCN [73]	59.2	63.0	66.1	77.5	66.5
Ours w/o AP	77.4	51.8	67.7	66.3	65.8
Ours w/ AP	**82.5**	**70.0**	**70.1**	**76.3**	**74.7**

In the second experiment we train our model on NTU dataset and test on MSR and UTD, which share 6 and 8 actions with NTU, respectively[2]. We only consider adaptation from NTU, which has a much large number of instances than MSR and UTD. The results are shown in Table 4. Similar to the first experiment, we observe consistent improvements by using AP in both datasets. The adaptation results are superior than the other three methods, which again shows the strength of Bayesian inference with AP.

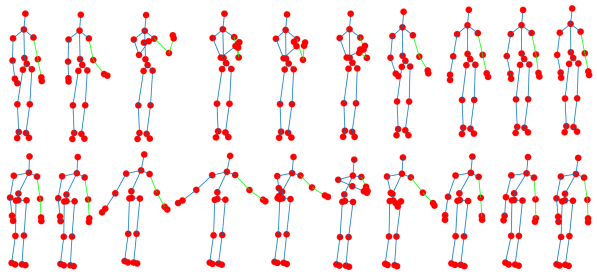

Figure 5. Examples of clapping. The first row is from a subject in UTD dataset. The second row is from a subject in MSR dataset.

4.5. Comparison with State-of-the-art

We compare the performance of action recognition with other state-of-the-art methods on the selected benchmark datasets. Despite being one of the earliest 3D action datasets, MSR Action 3D remains challenging due to noisy position caused by occlusion. There are many different evaluation protocols proposed in literatures. We use the cross-subject test, where the odd numbered subjects are used for training and the even numbered subjects are used

[1]MSR and UTD: *wave, catch, throw, draw x, draw circle, clap, jog, tennis swing, tennis serve, pick up & throw*.

[2]NTU and MSR: *throw, clap, cheer up, wave, kick, point*. NTU and UTD: *throw, sit down, stand up, clap, wave, point, cross hand, walk.*

for testing. According to Table 5, we outperform hand-crafted features based approaches [37]. We are also better than probabilistic model based approach [44], which improves the performance by combining hand-crafted features with probabilistic modeling. Our performance is comparable with recent deep learning based approaches [17, 45], despite using a much simpler design of RNN architecture.

Table 5. Classification results on MSR Action3D dataset.

Method	Accuracy
SC [37]	88.3
HBRNN [17]	94.5
Composition [44]	93.0
ST-LSTM [45]	**94.8**
Ours	94.5

For UTD dataset, we also use the odd/even split of the subjects for training and testing as suggested by the dataset authors. UTD includes several actions whose differences are subtle such as *drawing circle*, *drawing triangle* and *drawing x*. In Table 6, our approach outperforms both hand-crafted features based approaches [10, 7, 74] and deep learning based approach [33] by at least 5.1%, which shows the advantage of the proposed framework.

Table 6. Classification results on UTD dataset.

Method	Accuracy
Sensor Fusion [10]	79.1
DMM-LBP [7]	84.2
3DHoT-MBC [74]	84.4
SOS-CNN [33]	87.0
Ours	**92.1**

For SYSU dataset, we use the same cross-subject evaluation protocol as provided by the authors, where all the subjects are randomly and evenly split into training and testing set. SYSU is challenging due to the large number of subjects and each subject only performs each activity once, resulting large variations. The average number of frames per sequence is 203, which is also the longest among benchmarks we experimented with. From the results in Table 7, we show advantage over [45, 57] by better modeling of interdependencies among body joints through graph convolution. We are also better than [60], which selects a fixed size of subset of frames using reinforcement learning. Such strategy may alter the original dynamics of the action.

Table 7. Classification results on SYSU dataset.

Method	Accuracy
D-Skeleton [34]	75.5
ST-LSTM [45]	76.5
DPRL [60]	76.9
SR-TSL [57]	80.7
Ours	**82.0**

Finally, for NTU dataset, there are two different experiment settings. The first one is *cross subject*, where training and testing data contains different subjects without overlap. The second one is *cross view*, where training and testing data have different camera views. All the methods are NN based approaches due to their advantages on handling large-scale datasets. Compared to [17, 53, 47, 64], we show advantage due to two factors, the first is capturing dependencies among body parts and the second is improved generalization using adversarial prior. [73] also uses graph convolution, which treats time as another dimension of the graph. This strategy avoids using dynamic model such as RNN. But it requires a fixed length sequence. With a comparable performance, our model is more flexible in handling sequences with varied length. [60, 57] outperforms our approach, where [60] used a more sophisticated training scheme and [57] has a more sophisticated design of dynamic model. We believe our approach remains competitive given a simpler design of the dynamic model and strength in robustness and generalization.

Table 8. Classification results on NTU dataset.

Method	X-Subject	X-View
HBRNN [17]	59.1	64.0
PLSTM [53]	62.9	70.3
GLSTM [47]	74.4	82.8
2Stream-RNN [64]	71.3	79.5
ST-GCN [73]	81.5	88.3
DPRL [60]	83.5	89.8
ST-TSL [57]	**84.8**	**92.4**
Ours	81.8	89.0

5. Conclusion

In this paper, we address the challenge of modeling complex dynamics manifest in human motion data. We explicitly consider three sources of complexity including the interdependency among body joints during the motion, long-term temporal dependency among poses over the course of action, and subject-dependent variation. We propose a solution that combines graph convolution and LSTM to model spatio-temporal dynamics. The whole model is further extended to a probabilistic model following Bayesian framework with a novel adversarial prior. A Bayesian inference problem is formulated for classification in order to improve the robustness and generalization ability. Ablation study and evaluation on benchmark datasets show the effectiveness of the proposed framework.

Acknowledgment

This work is partially supported by Cognitive Immersive Systems Laboratory (CISL), a collaboration between IBM and RPI, and also a center in IBM's AI Horizon Network.

References

[1] Martin Abadi et al. Tensorflow: A system for large-scale machine learning. In *OSDI*, 2016. 5

[2] Justin Bayer and Christian Osendorfer. Learning stochastic recurrent networks. *arXiv*, 2014. 2, 4

[3] Yoshua Bengio, Yann LeCun, and Donnie Henderson. Globally trained handwritten word recognizer using spatial representation, convolutional neural networks, and hidden markov models. In *NIPS*, 1994. 2

[4] Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. *arXiv*, 2012. 2

[5] Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. Domain separation networks. In *NIPS*, 2016. 2

[6] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *ICLR*, 2014. 2

[7] Mohammad Farhad Bulbul, Yunsheng Jiang, and Jinwen Ma. Dmms-based multiple features fusion for human action recognition. *International Journal of Multimedia Data Engineering and Management*, 2015. 8

[8] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *CVPR*, 2017. 1

[9] Zhengping Che, Sanjay Purushotham, Guangyu Li, Bo Jiang, and Yan Liu. Hierarchical deep generative models for multi-rate multivariate time series. In *ICML*, 2018. 2

[10] Chen Chen, Roozbeh Jafari, and Nasser Kehtarnavaz. Utd-mhad: A multimodal dataset for human action recognition utilizing a depth camera and a wearable inertial sensor. In *ICIP*, 2015. 5, 8

[11] Tianqi Chen, Emily Fox, and Carlos Guestrin. Stochastic gradient hamiltonian monte carlo. In *ICML*, 2014. 5

[12] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv*, 2014. 6

[13] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C Courville, and Yoshua Bengio. A recurrent latent variable model for sequential data. In *NIPS*, 2015. 2, 4

[14] Hanjun Dai, Bo Dai, Yan-Ming Zhang, Shuang Li, and Le Song. Recurrent hidden semi-markov model. In *ICLR*, 2017. 2

[15] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *NIPS*, 2016. 2, 3

[16] Maxime Devanne, Hazem Wannous, Stefano Berretti, Pietro Pala, Mohamed Daoudi, and Alberto Del Bimbo. 3-d human action recognition by shape analysis of motion trajectories on riemannian manifold. *Cybernetics*, 2015. 2

[17] Yong Du, Wei Wang, and Liang Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *CVPR*, 2015. 1, 2, 8

[18] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. In *NIPS*, 2015. 2

[19] Meire Fortunato, Charles Blundell, and Oriol Vinyals. Bayesian recurrent neural networks. *arXiv*, 2017. 2, 4

[20] Marco Fraccaro, Søren Kaae Sønderby, Ulrich Paquet, and Ole Winther. Sequential neural models with stochastic layers. In *NIPS*, 2016. 2

[21] Yarin Gal and Zoubin Ghahramani. A theoretically grounded application of dropout in recurrent neural networks. In *NIPS*, 2016. 2

[22] Zhe Gan, Chunyuan Li, Changyou Chen, Yunchen Pu, Qinliang Su, and Lawrence Carin. Scalable bayesian learning of recurrent neural networks for language modeling. In *ACL*, 2017. 2, 4

[23] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. *arXiv*, 2014. 2

[24] Guillermo Garcia-Hernando and Tae-Kyun Kim. Transition forests: Learning discriminative temporal transitions for action recognition and detection. In *CVPR*, 2017. 2

[25] Andrew Gelman, Hal S Stern, John B Carlin, David B Dunson, Aki Vehtari, and Donald B Rubin. *Bayesian data analysis*. 2013. 4

[26] Dian Gong and Gerard Medioni. Dynamic manifold warping for view invariant action recognition. In *ICCV*, 2011. 2

[27] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014. 4

[28] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *ICASSP*, 2013. 4

[29] Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, and Daan Wierstra. Draw: A recurrent neural network for image generation. *arXiv*, 2015. 2

[30] David K Hammond, Pierre Vandergheynst, and Rémi Gribonval. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 2011. 2, 3

[31] Mikael Henaff, Joan Bruna, and Yann LeCun. Deep convolutional networks on graph-structured data. *arXiv*, 2015. 2

[32] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 3

[33] Yonghong Hou, Zhaoyang Li, Pichao Wang, and Wanqing Li. Skeleton optical spectra-based action recognition using convolutional neural networks. *IEEE Transactions on Circuits and Systems for Video Technology*, 2018. 8

[34] Jian-Fang Hu, Wei-Shi Zheng, Jianhuang Lai, and Jianguo Zhang. Jointly learning heterogeneous features for rgb-d activity recognition. In *CVPR*, 2015. 5, 8

[35] Mohamed E Hussein, Marwan Torki, Mohammad Abdelaziz Gowayyed, and Motaz El-Saban. Human action recognition using a temporal hierarchy of covariance descriptors on 3d joint locations. In *IJCAI*, 2013. 2

[36] Ashesh Jain, Amir R Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-rnn: Deep learning on spatio-temporal graphs. In *CVPR*, 2016. 2

[37] Min Jiang, Jun Kong, George Bebis, and Hongtao Huo. Informative joints based human action recognition using skeleton contexts. *Signal Processing: Image Communication*, 2015. 8

[38] Matthew Johnson, David K Duvenaud, Alex Wiltschko, Ryan P Adams, and Sandeep R Datta. Composing graphical models with neural networks for structured representations and fast inference. In *NIPS*, 2016. 2

[39] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *ICLR*, 2017. 2, 3

[40] Rahul G Krishnan, Uri Shalit, and David Sontag. Deep kalman filters. *arXiv*, 2015. 2

[41] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. *arXiv*, 2016. 2

[42] Maosen Li, Siheng Chen, Xu Chen, Ya Zhang, Yanfeng Wang, and Qi Tian. Actional-structural graph convolutional networks for skeleton-based action recognition. In *CVPR*, 2019. 2

[43] Wanqing Li, Zhengyou Zhang, and Zicheng Liu. Action recognition based on a bag of 3d points. In *CVPR Workshop*, 2010. 5

[44] Ivan Lillo, Juan Carlos Niebles, and Alvaro Soto. A hierarchical pose-based approach to complex action understanding using dictionaries of actionlets and motion poselets. In *CVPR*, 2016. 2, 8

[45] Jun Liu, Amir Shahroudy, Dong Xu, Alex Kot Chichung, and Gang Wang. Skeleton-based action recognition using spatio-temporal lstm network with trust gates. *TPAMI*, 2017. 8

[46] Jun Liu, Amir Shahroudy, Dong Xu, and Gang Wang. Spatio-temporal lstm with trust gates for 3d human action recognition. In *ECCV*, 2016. 2, 3

[47] Jun Liu, Gang Wang, Ping Hu, Ling-Yu Duan, and Alex C Kot. Global context-aware attention lstm networks for 3d action recognition. In *CVPR*, 2017. 8

[48] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *ECCV*, 2016. 1

[49] Siqi Nie, Ziheng Wang, and Qiang Ji. A generative restricted boltzmann machine based method for high-dimensional motion data modeling. *CVIU*, 2015. 2

[50] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. Learning convolutional neural networks for graphs. In *ICML*, 2016. 2

[51] Eshed Ohn-Bar and Mohan Trivedi. Joint angles similarities and hog2 for action recognition. In *CVPRW*, 2013. 2

[52] Liliana Lo Presti and Marco La Cascia. 3d skeleton-based human action classification: A survey. *Pattern Recognition*, 2016. 2

[53] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. Ntu rgb+ d: A large scale dataset for 3d human activity analysis. In *CVPR*, 2016. 5, 8

[54] Nino Shervashidze, SVN Vishwanathan, Tobias Petri, Kurt Mehlhorn, and Karsten Borgwardt. Efficient graphlet kernels for large graph comparison. In *AISTATS*, 2009. 2

[55] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In *CVPR*, 2019. 2

[56] Jamie Shotton, Toby Sharp, Alex Kipman, Andrew Fitzgibbon, Mark Finocchio, Andrew Blake, Mat Cook, and Richard Moore. Real-time human pose recognition in parts from single depth images. *Communications of the ACM*, 2013. 1

[57] Chenyang Si, Ya Jing, Wei Wang, Liang Wang, and Tieniu Tan. Skeleton-based action recognition with spatial reasoning and temporal stack learning. *ECCV*, 2018. 1, 8

[58] Sijie Song, Cuiling Lan, Junliang Xing, Wenjun Zeng, and Jiaying Liu. An end-to-end spatio-temporal attention model for human action recognition from skeleton data. In *AAAI*, 2017. 2

[59] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *ECCV*. Springer, 2016. 2

[60] Yansong Tang, Yi Tian, Jiwen Lu, Peiyang Li, and Jie Zhou. Deep progressive reinforcement learning for skeleton-based action recognition. In *CVPR*, 2018. 2, 3, 8

[61] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *CVPR*, 2017. 2

[62] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv*, 2014. 2

[63] Raviteja Vemulapalli, Felipe Arrate, and Rama Chellappa. R3dg features: Relative 3d geometry-based skeletal representations for human action recognition. *CVIU*, 2016. 2, 7

[64] Hongsong Wang and Liang Wang. Modeling temporal dynamics and spatial configurations of actions using two-stream recurrent neural networks. In *CVPR*, 2017. 2, 8

[65] Hongsong Wang and Liang Wang. Beyond joints: Learning representations from primitive geometries for skeleton-based action recognition and detection. *TIP*, 2018. 2

[66] Jiang Wang, Zicheng Liu, Ying Wu, and Junsong Yuan. Mining actionlet ensemble for action recognition with depth cameras. In *CVPR*, 2012. 2

[67] Pichao Wang, Zhaoyang Li, Yonghong Hou, and Wanqing Li. Action recognition based on joint trajectory maps using convolutional neural networks. In *Multimedia*, 2016. 3

[68] Pei Wang, Chunfeng Yuan, Weiming Hu, Bing Li, and Yanning Zhang. Graph based skeleton motion representation and similarity measurement for action recognition. In *ECCV*, 2016. 2

[69] Xishun Wang, Minjie Zhang, and Fenghui Ren. Sparse gaussian conditional random fields on top of recurrent neural networks. 2018. 2

[70] Junwu Weng, Mengyuan Liu, Xudong Jiang, and Junsong Yuan. Deformable pose traversal convolution for 3d action and gesture recognition. In *ECCV*, 2018. 2, 3

[71] Junwu Weng, Chaoqun Weng, and Junsong Yuan. Spatio-temporal naive-bayes nearest-neighbor (st-nbnn) for skeleton-based action recognition. In *CVPR*, 2017. 2

[72] Di Wu and Ling Shao. Leveraging hierarchical parametric networks for skeletal joints based action segmentation and recognition. In *CVPR*, 2014. 2

[73] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *AAAI*, 2018. 2, 7, 8

[74] Baochang Zhang, Yun Yang, Chen Chen, Linlin Yang, Jungong Han, and Ling Shao. Action recognition using 3d histograms of texture and a multi-class boosting classifier. *TIP*, 2017. 8

[75] Jing Zhang, Wanqing Li, Philip O Ogunbona, Pichao Wang, and Chang Tang. Rgb-d-based action recognition datasets: A survey. *Pattern Recognition*, 2016. 2

[76] Xikang Zhang, Yin Wang, Mengran Gou, Mario Sznaier, and Octavia Camps. Efficient temporal sequence comparison and classification using gram matrix embeddings on a riemannian manifold. In *CVPR*, 2016. 2

[77] Rui Zhao, Wanru Xu, Hui Su, and Qiang Ji. Bayesian hierarchical dynamic model for human action recognition. In *CVPR*, 2019. 2

[78] Wentao Zhu, Cuiling Lan, Junliang Xing, Wenjun Zeng, Yanghao Li, Li Shen, Xiaohui Xie, et al. Co-occurrence feature learning for skeleton based action recognition using regularized deep lstm networks. In *AAAI*, 2016. 2, 7