IEEE Signal Processing Society
IEEE Open Journal of
Signal Processing

Received 12 December 2023; revised 14 April 2024; accepted 22 April 2024. Date of publication 3 May 2024;
date of current version 23 September 2024. The review of this article was arranged by Associate Editor Lisandro Lovisolo.

Digital Object Identifier 10.1109/OJSP.2024.3396635

Short Paper

Towards a Geometric Understanding of Spatiotemporal Graph Convolution Networks

PRATYUSHA DAS, SARATH SHEKKIZHAR, AND ANTONIO ORTEGA (Fellow, IEEE)

University of Southern California, Los Angeles, CA 90089 USA

CORRESPONDING AUTHOR: PRATYUSHA DAS (e-mail: daspraty@usc.edu).

This article has supplementary downloadable material available at https://doi.org/10.1109/OJSP.2024.3396635, provided by the authors.

ABSTRACT Spatiotemporal graph convolutional networks (STGCNs) have emerged as a desirable model for *skeleton*-based human action recognition. Despite achieving state-of-the-art performance, there is a limited understanding of the representations learned by these models, which hinders their application in critical and real-world settings. While layerwise analysis of CNN models has been studied in the literature, to the best of our knowledge, there exists *no study* on the layerwise explainability of the embeddings learned on spatiotemporal data using STGCNs. In this paper, we first propose to use a local Dataset Graph (DS-Graph) obtained from the feature representation of input data at each layer to develop an understanding of the layer-wise embedding geometry of the STGCN. To do so, we develop a window-based dynamic time warping (DTW) method to compute the distance between data sequences with varying temporal lengths. To validate our findings, we have developed a layer-specific Spatiotemporal Graph Gradient-weighted Class Activation Mapping (L-STG-GradCAM) technique tailored for spatiotemporal data. This approach enables us to visually analyze and interpret each layer within the STGCN network. We characterize the functions learned by each layer of the STGCN using the label smoothness of the representation and visualize them using our L-STG-GradCAM approach. Our proposed method is generic and can yield valuable insights for STGCN architectures in different applications. However, this paper focuses on the human activity recognition task as a representative application. Our experiments show that STGCN models learn representations that capture general human motion in their initial layers while discriminating different actions only in later layers. This justifies experimental observations showing that fine-tuning deeper layers works well for transfer between related tasks. We provide experimental evidence for different human activity datasets and advanced spatiotemporal graph networks to validate that the proposed method is general enough to analyze any STGCN model and can be useful for drawing insight into networks in various scenarios. We also show that noise at the input has a limited effect on label smoothness, which can help justify the robustness of STGCNs to noise.

INDEX TERMS STGCN, NNK, KNN, geometric interpretation, graph neural network, transfer learning.

I. INTRODUCTION

Deep learning models have led to significant advances in application domains, such as images and video [1], [2], where data is available on a regular grid, e.g., formed by pixels. More recently, graph neural networks (GNNs) [3], and graph convolutional networks (GCNs) [4] have been proposed to handle *data with irregular structures*, such as social networks [5], skeleton-based motion capture data (MoCap) [6]. In this paper, we focus on spatiotemporal graph convolutional networks (STGCNs). STGCNs can efficiently handle the temporal aspects of graph data and have wide-ranging applications, including in tasks such as traffic forecasting [7] and the recognition of actions based on skeletal data [8], [9].

When it comes to training STGCN models, there are several crucial design choices to consider, such as the architecture, optimization routine, loss function, and dataset. Usually, these choices interplay in intricate ways to shape the characteristics of the final model. Therefore, selecting a particular model is often primarily based on its *performance* on specific datasets. While this practical perspective has led to significant advances, achieving a deeper understanding of the system is essential for ensuring safe and robust real-world deployment.

Two major approaches have been used in the literature to understand deep learning systems. *Function approximation* methods are based on the inductive bias of the loss function [10], the ability of

the optimization to achieve good minima [11], or consider the study of classifier margins [12]. *Data-driven* analysis methods consider the relative position of sample data points in the representation domain for characterization [13], [14], [15]. Data-driven approaches can provide a unified framework for understanding models because they can abstract the specific functional components. Specifically, in a data-driven approach, functions do not need to be explicitly modeled; they can be characterized implicitly using the outputs they produce.

In this paper, we develop a *data-driven* approach to achieve a better understanding of STGCN models. Our approach is based on a layer-wise analysis, interpretation, and visualization of the embeddings produced by the STGCN. Our proposed method starts by defining a **Dataset graph (DS-Graph)**, which captures the pairwise similarities between sequences in the set, represented by their embedding. This allows us to compare models obtained with very different architectures by simply comparing the DS-Graphs they produce in their respective embedded spaces. While our method is widely applicable, our experiments focus on a human activity recognition task using skeleton-based data as an illustrative task to evaluate our STGCN analysis methods. This type of data has been widely used in human action recognition due to its view-invariant representation of pose structure, robustness to sensor noise [16], and efficiency in computation and storage [17], [18]. Recently, STGCN approaches have gained popularity by demonstrating superior performance in human activity understanding [19], [20], [21], [22] and have become one of the state-of-the-art methods in the field of activity recognition. As will be shown experimentally, our proposed layer-wise analysis of STGCNs helps us to (i) understand their generalization, (ii) detect bias toward learning any particular feature, (iii) evaluate model invariance to a set of functions, and (iv) assess robustness to perturbations to the input data. For example, in STGCN for skeleton-based activity recognition [8], some layers may focus on learning the motion of specific body parts. Therefore, some models will not be suitable for new action classes where the motion is localized in other body parts.

While layer-wise analysis of CNN models [23] and feature visualization methods [24], [25] have been studied in the literature [1], [2], to the best of our knowledge, there exists *no study* on the layer-wise explainability of the embeddings learned on spatiotemporal data using STGCNs. Moreover, layer-wise feature visualization techniques for STGCNs are also not available. In fact, most of the work on STGCN interpretation has studied only the final layer [19], [26]. Extending the layer-wise analysis to STGCNs is not straightforward because of the varying lengths of the STGCN embeddings. This variability makes it difficult to find the similarity of embeddings of data points, such as action sequences with differing lengths, as the commonly employed similarity metrics (e.g., cosine similarity or Euclidean distance) are unsuitable for sequences of varying lengths.

Our first major contribution is a *geometric* framework to characterize the data manifolds corresponding to each STGCN layer output. Our approach analyzes these manifolds by constructing a Non-Negative Kernel (NNK) DS-Graph [27] (Section II-C), where nodes represent input sequences (actions) and distances between nodes are computed using dynamic time warping (DTW) [28] (Section II-D). This allows a distance to be computed between actions with different durations. We choose the NNK construction due to its robust performance in local estimation across different machine learning tasks [14]. The benefits of the NNK construction will be demonstrated through a comparison with k-NN DS-Graph constructions in Section IV-B.

For the DS-Graph at each layer, we quantify the label smoothness as a way to track how the STGCN learns (Fig. 1).

Our approach has several important advantages: (1) the analysis is agnostic to the training procedure, architecture, or loss function used to train the model; (2) it allows for the comparison of features having different dimensions; (3) it can be applied to data that were not used for training (e.g., unseen actions or data in a transfer setting); (4) it allows us to observe how the layerwise representations are affected by external noise added to the input.

Our second major contribution is to extend our previous method, spatiotemporal graph GradCAM (STG-GradCAM) [26], to perform layerwise visualization of the contributions of different Skeleton-Graph (S-Graph) nodes. To achieve this, we merge the class-specific gradient for a datapoint at each layer with the learned representations by that layer. This enables us to interpret individual layers within an STGCN network. The resulting layerwise STG-GradCAM (L-STG-GradCAM) allows us to visualize the importance of any node in any STGCN layer for the classification of a particular query class (action). This visualization helps confirm the results obtained through our analysis of the STGCN model using NNK-based geometric methods. It enhances the transparency of the model and deepens our comprehension of the representations learned at each layer. With our proposed data-driven label smoothness and layerwise visualization from L-STG-GradCAM, we can show that: (1) Initial layers learn low-level features corresponding to general human motion, while specific actions are recognized only in the later layers. (2) In a transfer task, the choice of which layers to leave unchanged and which layers to fine-tune can be informed by the changes in label smoothness for the target task on a network trained for the source task. (3) Experimentally, the label smoothness of an STGCN model over the layers as measured in the dataset graph is not affected significantly when Gaussian noise is added to the inputs, which justifies the observation that the model is robust to noise.

II. PRELIMINARIES

A. SKELETON GRAPH AND POLYNOMIAL GRAPH FILTERS

A skeleton graph (S-Graph) is a fixed undirected graph $\mathcal{G}_S = \{\mathcal{V}, \mathcal{E}, \hat{\mathbf{A}}\}$ composed of a vertex set $\mathcal{V}$ of cardinality $|\mathcal{V}| = N$, an edge set $\mathcal{E}$ connecting vertices, and $\hat{\mathbf{A}}$, a weighted adjacency matrix. $\hat{\mathbf{A}}$ is a real symmetric $N \times N$ matrix, where $a_{i,j} \geq 0$ is the weight assigned to the edge connecting nodes i and j. An STGCN layer (Section II-B) is a function of this adjacency matrix $\hat{\mathbf{A}}$ and the identity matrix $\mathbf{I}$ representing a self-loop. Specifically, STGCN uses the normalized adjacency matrix $\mathbf{A} = \mathbf{D}^{-\frac{1}{2}}(\hat{\mathbf{A}} + \mathbf{I})\mathbf{D}^{-\frac{1}{2}}$ where $(\mathbf{D})_{ii} = (\sum_j a_{i,j}) + 1$. Intuitively, the elementary graph filter $\mathbf{A}$ combines graph signals from adjacent nodes. Self-loops are added so that a node's own features are combined with those of its neighbors for learning.

Fig. 2 provides an example of how human motion (in this case, we consider a skeleton graph with 25 nodes) is projected onto the eigenvectors of $\mathbf{A}$, leading to energy that is typically concentrated in the eigenvectors corresponding to the larger eigenvalues of $\mathbf{A}$ (i.e., $\lambda_{17}, \ldots, \lambda_{25}$).[1] In each layer, we use simple filters of the form $\boldsymbol{x}_{out} = \mathbf{A}\boldsymbol{x}_{in}\mathbf{W}$, where $\mathbf{W}$ are trainable weights. Applying these simple one-hop filters in multiple successive layers allows us to learn over multi-hop graph neighborhoods, analogous to what

[1]Note that the larger eigenvalues of $\mathbf{A}$ correspond to the smaller eigenvalues of the graph Laplacian $\mathbf{I} - \mathbf{A}$. Thus, energy concentration in the higher eigenvalues of $\mathbf{A}$ shows that typical human motion is smooth.

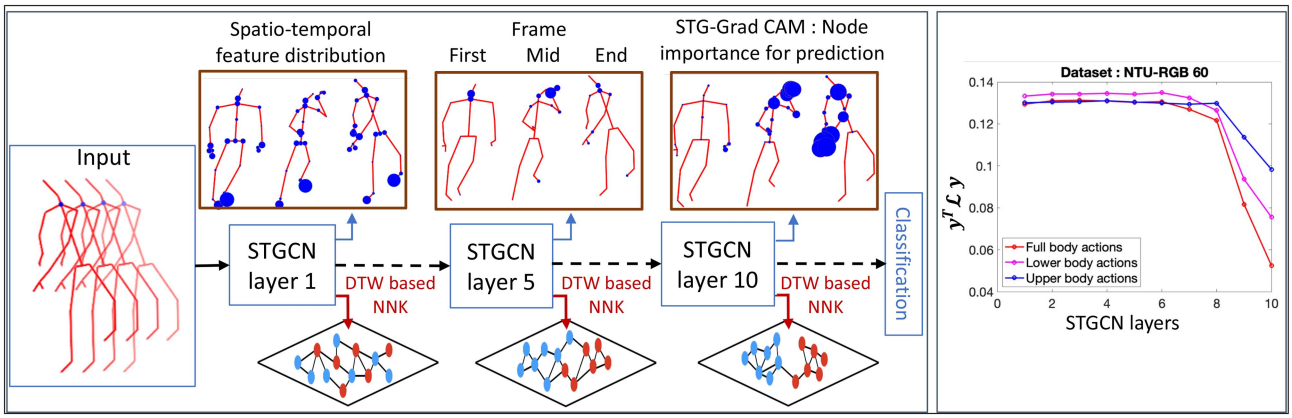

FIGURE 1. Proposed data-driven approach to understanding the geometry of the embedding manifold in STGCNs using windowed dynamic time warping (DTW) and non-negative kernel (NNK) graphs. **Left:** We construct *dataset NNK Graphs (DS-Graph)* where each node corresponds to an action sequence, and the weights of edges connecting two nodes are derived from pairwise distances between the features representing the corresponding action sequences. In this example, we show how the two classes (corresponding to red and blue nodes on the DS-Graph) become more clearly separated in deeper layers of the network. We also observe the *skeleton graph (S-Graph)* node importance for each action using a layerwise STG-GradCAM (the three-time slice example corresponds to a *Throw* action). **Right:** For a set of spatiotemporal input action sequences, we observe the label smoothness on the *DS-Graph* constructed using the features obtained for the sequences after each STGCN layer. The observed label smoothness at each layer of the STGCN network averaged over three super-classes corresponding to actions involving the upper body, lower body, and full body. In this plot, lower variation corresponds to greater smoothness. We note that the label smoothness increases in the deeper layers, in which the different actions can be classified (see DS-Graphs at the bottom of the left plot).

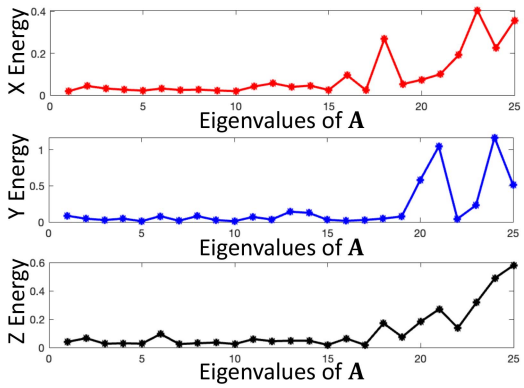

FIGURE 2. Energy graph spectrum of the human actions (NTURGB 120 [30]) of the normalized adjacency matrix of the S-Graph (A). We use the graph spectrum of the adjacency matrix, as used in the STGCN, for ease of understanding.

could be achieved with higher-order polynomials of the adjacency matrix,where an l-degree polynomial captures the data in a l-hop neighborhood. For example, in the human activity recognition task, the S-Graph has 25 nodes and 24 edges [29]. For this tree-structured graph, the maximum distance between two leaf nodes is 10. Thus, a 10-degree polynomial can capture information about the entire graph. This justifies using at most 10 layers of STGCN units in the STGCN network under consideration, where each layer is a function of $\mathbf{A}$, i.e., a polynomial of degree 1.

B. SPATIOTEMPORAL GRAPH CONVOLUTIONAL NETWORK

STGCN for action recognition was first adopted in [8], where the spatial graph represented the intra-body connections of joints. While considering a spatiotemporal signal $\boldsymbol{x}$, the input feature is represented as a $C \times N \times T$ tensor, where C, N, and T represent the number of channels, number of joints, and the temporal length of the activity sequence, respectively. Graph convolutions are performed in two stages. First, a convolution is performed with a temporal kernel of size $(1 \times \tau)$. Second, to capture the intra-joint variations, the resulting tensor is multiplied by the normalized adjacency matrix $\mathbf{A}$ along the spatial axis. Denoting the input and output features of an STGCN layer as $\boldsymbol{x}_{in}$ and $\boldsymbol{x}_{out}$, the STGCN mapping is given by $\boldsymbol{x}_{out} = \mathbf{A}\boldsymbol{x}_{in}\mathbf{W}$, where $\mathbf{W}$ represents trainable weight tensor corresponding to multiple input channels. Another matrix $\mathbf{Q}$ is introduced to learn the edge weights of the graph. Thus, each STGCN layer is implemented as follows:

$$\boldsymbol{x}_{out} = \sum_j \mathbf{D}_j^{-\frac{1}{2}}(\mathbf{A}_j \otimes \mathbf{Q})\mathbf{D}_j^{-\frac{1}{2}}\boldsymbol{x}_{in}\mathbf{W}_j, \tag{1}$$

where $\otimes$ denotes the Hadamard product.

C. NON-NEGATIVE KERNEL(NNK) NEIGHBORHOODS

We use non-negative kernel regression (NNK) neighborhoods and graphs [31] for our manifold analysis because this results in better neighborhood construction with improved and robust local estimation performance in various machine learning tasks [14], [32]. The key advantage of NNK is its geometric interpretation for each neighborhood constructed. While in KNN points $\mathbf{x}_j$ and $\mathbf{x}_k$ are included in the neighborhood of a data point $\mathbf{x}_i$ solely based on their similarity to $\mathbf{x}_i$, i.e., $s(\mathbf{x}_i, \mathbf{x}_j)$ and $s(\mathbf{x}_i, \mathbf{x}_k)$, in NNK this decision is made by also taking into account the metric $s(\mathbf{x}_j, \mathbf{x}_k)$. Consequently, $\mathbf{x}_j$ and $\mathbf{x}_k$ are both included in the NNK neighborhood only if they are not geometrically *redundant*, the details are given in (II-C). NNK uses KNN as an initial step, with only a modest additional runtime requirement [31]. The computation can be accelerated using tools [33] developed for KNN when dealing with large datasets. NNK requires kernels with a [0, 1] range. In this work, we use the cosine similarity with the windowed aggregation in (3). This kernel is applied to representations obtained after ReLU and satisfies the NNK definition requirement.

D. DYNAMIC TIME WARPING

While Euclidean distance permits only one-to-one point comparison, Dynamic Time Warping (DTW) (Section II-C) accommodates many-to-one comparisons, allowing precise alignment while considering temporal variations. In our action recognition task, we work with action sequences of different durations, and our STGCN-based feature extraction retains temporal information. In this study, we employ DTW to measure the similarity between temporal features extracted by STGCN. DTW is computed using 2, where dtw(i, j) represents the minimum warp distance between two time series of lengths i and j. Each element in the accumulated matrix reflects the DTW distance between series $U_{1:i}$ and $V_{1:j}$.

$$\begin{aligned}\mathrm{DTW}(i, j) = \mathrm{dist}(u_i, v_j) + \min(\mathrm{DTW}(i-1, j),\\ \mathrm{DTW}(i, j-1), \mathrm{DTW}(i-1, j-1))\end{aligned} \tag{2}$$

III. PROPOSED GEOMETRIC ANALYSIS OF STGCN

A. NEIGHBORHOOD ANALYSIS USING DYNAMIC DTW

Once we have a fully trained STGCN network, we construct an NNK DS-Graph using the representation generated by each layer of the STGCN model and refer to this graph as the NNK *NNK Dataset Graph* $\mathcal{G}_D$. Note that each node corresponds to a data point in our *NNK DS-Graph*, i.e., an action sequence represented by its features (learned by the STGCN). This differs from the S-Graph used in the STGCN model, which provides the original representation of an action sequence from which the features are extracted. After *NNK DS-Graph* construction, we observe the smoothness of the class labels with respect to the graph, as shown in Fig. 1. Graph smoothness or label smoothness in a graph represents the variation of the label of the neighboring node for each node in the DS-Graph. A DS-Graph has higher label smoothness when there is less variation in the labels of neighboring nodes. Our work uses label smoothness as a metric for assessing the representation of different layers within a network.

The main challenge with spatiotemporal action data is that each individual activity corresponds to a data sequence with a different temporal length.

To address this issue, we develop a DTW-based distance metric to find the similarity between the representations (Section II-D). Computation of this *window-based DTW* distance metric *w-DTW* involves the following steps.

- Consider two sequences $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$ divided temporally into m windows. The dimension of $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$ is $N \times T_i$ and $N \times T_j$ respectively. Here N denotes the number of spatial joints and T_i denotes the temporal length of the ith sequence.
- $\boldsymbol{s}_i^w$ denotes the w-th window of the sequence, then the distance between two sequences is computed as follows.

$$\mathrm{wDTW}(\boldsymbol{s}_i, \boldsymbol{s}_j) = \sum_{w=1}^{m} \alpha_w \mathrm{DTW}(\boldsymbol{s}_i^w, \boldsymbol{s}_j^w) \tag{3}$$

  α_w is the weight to the w-th window, $\sum_{w=1}^{m} \alpha_w = 1$.
- The weights are chosen such that they decrease along the temporal axis based on the length statistic of all the sequences in the dataset i.e., the number of samples that have non-zero padding in a particular temporal window.

While STGCN involves complex mappings, the transformations they induce and the corresponding structure of each representation space can be studied using a graph constructed on the embedded features. Consider an STGCN model and a spatiotemporal dataset. At each layer, all sequences in the dataset can be represented using the *NNK Dataset Graph*. In this graph, each node corresponds to a sequence, and the action labels are treated as 'signals' or attributes associated with these nodes, as illustrated in Fig. 1. At the output of each layer, each input sequence is mapped to new values (in some other feature space). Thus, we can associate a new NNK DS-graph to the same set of data points (with the same signal, i.e., label). Thus, instead of directly working with the high dimensional features or the model's overparameterized space, the focus is on the relative positions of the feature embeddings obtained in STGCN layers. This allows us to characterize the geometry of the manifold spaces encoded by an STGCN and to develop a quantitative understanding of the model.

We now present a theoretical result (Theorem 1) relating the respective label smoothnesses of the input and output features of a single layer in a neural network to that of its complexity measured by the ℓ_2-norm [12], [34]. The proof for the theorem is provided in the supplementary materials (Section VI.B).

Definition 1 (Label smoothness): Given a graph represented by its Laplacian $\mathcal{L}$ and a label signal $\mathbf{y}$ on the graph, the Laplacian quadratic $\mathbf{y}^\top \mathcal{L}\mathbf{y}$ captures the smoothness of the label on the graph [35], [36]. Note that smaller values of $\mathbf{y}^\top \mathcal{L}\mathbf{y}$ correspond to smoother signals. In other words, an increase in the label similarity of the connected nodes is commensurate with a decrease in $\mathbf{y}^\top \mathcal{L}\mathbf{y}$.

Theorem 1: Consider the features corresponding to the input and output of a layer in a neural network denoted by $\boldsymbol{x}_{out} = \phi(\mathbf{W}\boldsymbol{x}_{in})$ where $\phi(\boldsymbol{x})$ is a slope restricted nonlinearity applied along each dimension of $\boldsymbol{x}$. Let us suppose that the smoothness of the labels $\boldsymbol{y}$ in the feature space is proportional to the smoothness of the data $\boldsymbol{x}$. Then,

$$\mathbf{y}^\top \mathcal{L}_{out}\mathbf{y} \leq c\ ||\mathbf{W}||_2^2\ \ \mathbf{y}^\top \mathcal{L}_{in}\mathbf{y} \tag{4}$$

where $\mathcal{L}$ corresponds to the graph laplacian obtained using NNK in the feature space. Note that $c > 0$ depends only on constants related to data smoothness and the slope of the non-linearity.

Remark 1: Theorem 1 states that the change in label smoothness between the input and output spaces of a network layer is indicative of the complexity of the mapping induced by that layer, i.e., a big change in label smoothness corresponds to a larger transformation of the features space.

Remark 2: Theorem 1 does not make any assumption on the model architecture and makes an assumption about the relationship between the respective smoothness of the data and the labels. The slope restriction on the nonlinearity is satisfied by activation functions used often in practice. For example, the ReLU function is slope restricted between 0 and 1 [37], [38].

The idea of characterizing intermediate representations using graphs was previously studied in [39], [40]. However, these works were limited to images and did not study spatiotemporal data. To the best of our knowledge, our work presents the first method for use with structured input sequences for analysis and understanding of STGCN networks.

Our method uses NNK for analysis similar to [23]. However, unlike other approaches, our work focuses on the geometry of the feature manifold induced by the STGCN layer using the NNK graphs constructed.

B. LAYERWISE (L) STG-GRADCAM

As in regular convolutional layers, unlike fully-connected layers, spatiotemporal graph convolution layers retain localized information both in the spatial and temporal axis. [26] proposed STG-GradCAM for visualizing the importance of the nodes in the spatiotemporal

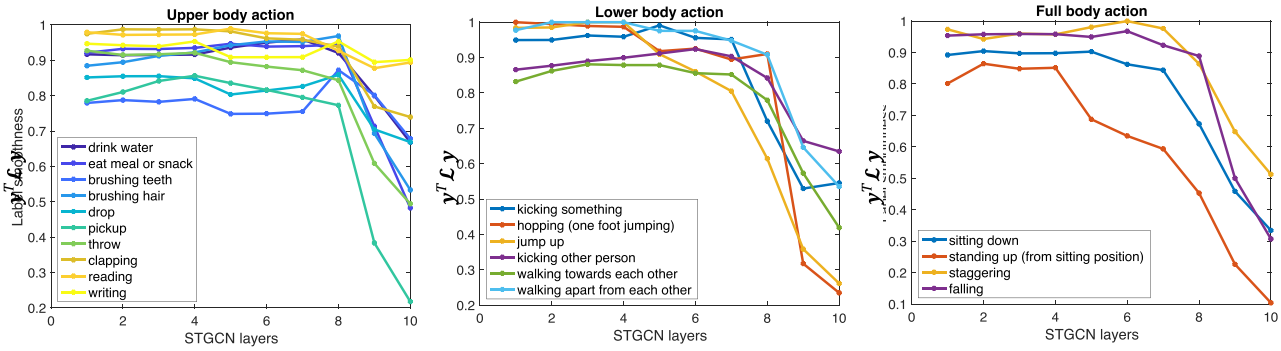

FIGURE 3. **Smoothness of labels on the manifold induced by the STGCN layer mappings in a trained model. As the label smoothness increases, the Laplacian quadratic ($\mathbf{y}^\top \mathcal{L}\mathbf{y}$) decreases. Intuitively, a lower value of $\mathbf{y}^\top \mathcal{L}\mathbf{y}$ corresponds to the features belonging to a particular class having graph neighbors from the same class.We divide the actions in NTU-RGB60 into three super-classes (Upper body (Left), Lower body (Middle), Full body (Right)) and present smoothness with respect to each action in the grouping. We emphasize that, though the smoothness is displayed per class, the *NNK Dataset Graph* is constructed using the features corresponding to all input action data points. We observe that the model follows a similar trend, where the smoothness of labels is flat in the initial layers (indicative of no class-specific learning) and increases in value in the later layers (corresponding to discriminative learning). Outliers exist to this trend (e.g., in upper body group *drop, brushing*) where the smoothness decreases in intermediate layers. This may imply that the representations for these actions are affected by features from other actions to accommodate for learning other classes.**

skeleton graph for a particular action. However, that work only used the last STGCN layer to provide an interpretation. In this paper, we extended STG-GradCAM to L-STG-GradCAM for use with all layers of an STGCN model.

The gradient information flowing into each STGCN layer is used in our proposed L-STG-GradCAM to compute the importance of each neuron for a particular class prediction and to determine whether the intermediate layers are learning something meaningful. We use the gradients as the weight of the representations at each layer. The outcome of L-STG-GradCAM helps us to understand which part of the data in each layer contributes to the final decision. Let the kth graph convolutional feature map at layer ℓ be defined as: $\mathbf{F}_k^\ell(\mathbf{X}, \mathbf{A}) = \sigma(\tilde{\mathbf{A}}\mathbf{F}^{\ell-1}(\mathbf{X}, \mathbf{A})\mathbf{W}_k^\ell)$. Here, $\tilde{\mathbf{A}} = (\mathbf{D}^{-\frac{1}{2}}(\mathbf{A}+\mathbf{I})\mathbf{D}^{-\frac{1}{2}}) \odot \mathbf{Q}$.

Here, the kth feature at the ℓth layer is denoted by $\mathbf{F}_{k,n,t}^\ell$ for node n and time t. Then, L-STG-GradCAM's label-specific weights for class c at layer l and for feature k are calculated by:

$$\boldsymbol{\beta}_k^{c,\ell} = \frac{1}{NT}\sum_{n=1}^{N}\sum_{t=1}^{T}\frac{\delta y^c}{\delta \mathbf{F}_{k,n,t}^\ell}. \tag{5}$$

Here, y_c is the cth class score. Then, we can compute the importance of the nodes in a specific layer ℓ using:

$$H_{ST}^{c,\ell} = \text{ReLU}\left(\sum_k \boldsymbol{\beta}_k^{c,\ell}\mathbf{F}_k^\ell\right). \tag{6}$$

L-STG-GradCAM enables us to visualize the class-specific spatiotemporal importance (Fig. 5) of the representation for any layer ℓ of the network. The code is https://github.com/daspraty/stg-gradcam.gitavailable.

IV. RESULTS

A. EXPERIMENT SETTING

1) NETWORK ARCHITECTURE

The STGCN model used for human action recognition by [8] comprises 10 STGCN layers implemented as in Section II-B. The first four layers have 64 output channels, the next three layers have 128, and the last three have 256. Afterward, a global pooling layer with a softmax is used as a classifier. The model is trained using cross-entropy loss with batch SGD for 100 epochs.

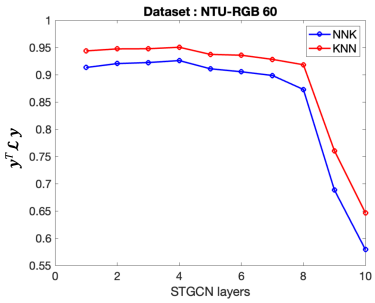

FIGURE 4. **Label smoothness of STGCN for different DS-graph construction methods (blue)-NNK, (red)-*k*-NN.**

2) DATASETS: NTU-RGB60

We use the STGCN model described above and introduced in [8] and train the model in cross-subject (x-sub) settings on NTU-RGB60 [29] dataset. This dataset contains 56,000 action clips corresponding to 60 action classes performed by 40 subjects, e.g., Throw, Kick). The dataset includes annotated 3D joint locations (X, Y, Z) of 25 joints. *NTU-RGB120:* NTU-RGB120 [30] extends NTU-RGB60 with an additional 57,367 skeleton sequences over 60 extra action classes, from 106 distinct subjects.

B. LABEL SMOOTHNESS COMPUTED FROM THE FEATURES

To provide insights into the representations of the intermediate layers of the STGCN network, we make use of our geometric analysis of the representation using the DTW-based NNK method described in Section III. Fig. 1 (right) shows the label smoothness over the layers of STGCN for different sets of the upper body, lower body, and full body actions (refer to Table 3). We see a sudden fall in the Laplacian quadratic after layer 8, while the slope is small before that layer. This implies that the early layers have features that are mostly not class-specific. In contrast, the smoothness improves (corresponds to a decrease in the value of $\mathbf{y}^\top \mathcal{L}_{out}\mathbf{y}$) using the representations after layer 8 consistently across all input actions. Following Theorem 1,

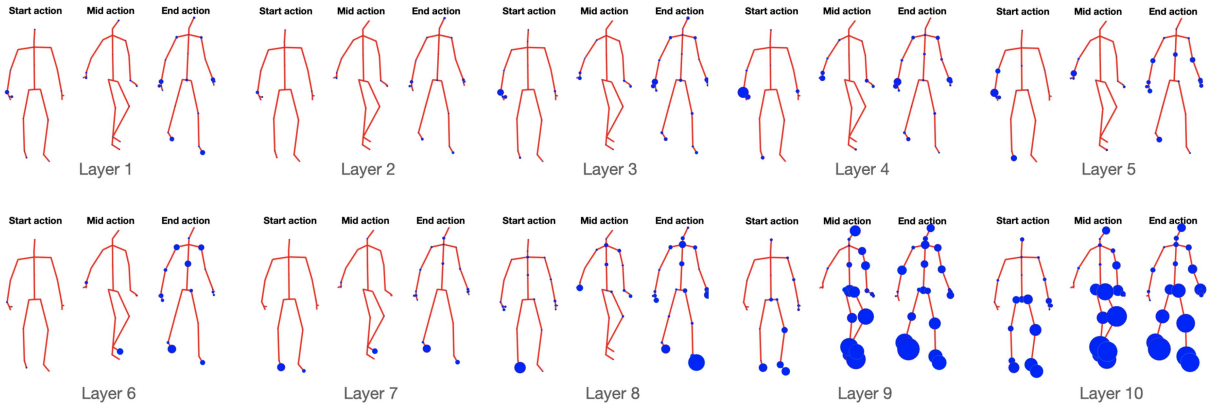

FIGURE 5. **L-STG-GradCAM visualization of spatiotemporal node importance for action class *Kick* of a trained STGCN network used in experiments. The size of the blue bubble denotes the relative importance of the node in a layer for prediction by the final softmax classifier and is scaled to have values in [0, 1] at each layer. The node importance values are normalized across layers to have a clear comparison among the layers. We observe that the localization of the action as observed using the L-STG-GradCAM is evident only in later layers while initial layers have no class-specific influence. The visualizations allow for transparency in an otherwise black-box model to explain any class prediction. Our approach is applicable to any STGCN model and is not affected by the model size, optimization strategy, or dataset used for training.**

we can state that the large change in the label smoothness from layer 8 to 9 corresponds to a larger transformation (equivalent to the functional norm of the layer is large) in the input-output mapping of this layer. Our earlier visualization using L-STG-GradCAM validates this analysis visually. Fig. 3 presents action-wise label smoothness over the layers of STGCN. This figure helps us better understand which actions are learned over the layer of STGCN. For example, in Fig. 3 (left), action *reading* and *writing* are poorly learned. The value of $\mathbf{y}^\top \mathcal{L}_{out}\mathbf{y}$ in the plot is not monotonically decreasing for all the actions. For example, for the action *drop*, the label smoothness decreases in the middle and increases again at the end. The possible reason behind this pattern is that the network tries to accommodate other actions and again learns all the actions gradually before the last layer.

Comparison between NNK and k-NN: Fig. 4 shows the effect on label smoothness for different choices of graph construction methods like NNK and k-NN. Higher label smoothness (small value of $\mathbf{y}^\top \mathcal{L}_{out}\mathbf{y}$) represents better the construction of the graph, reducing the prediction error at each layer. NNK clearly performs better than k-NN in choosing the right neighbors and their corresponding weights.

C. L-STG-GRADCAM VISUALIZATION

The STGCN model in Section IV-A with NTU-RGB60 achieves 82.1% accuracy on NTU-RGB60 *xsub* setting [8]. We use (6) to generate a class-specific skeleton joint-time importance map using all the data points corresponding to a given class. Fig. 5 shows the layerwise variation of joint importance for the action *'Kick'* for three time-slices. The node's size in the S-graph denotes the degree of importance of the body joint at that time point for the final prediction. For the action *Kick*, which mostly involves lower body parts, we notice in the figure that the initial layers (up to layer 8) have very weak, if any, GradCAM localization corresponding to the action. In contrast, the last three STGCN layers show explicit node importance *heatmap* where the leg and the back joints are relatively more active, indicative of the action. We present additional examples in the supplementary (Fig. 12, Fig. 13) corresponding to an upper-body action (*Throw*) and a full body action (*Sitting down*). In both cases, we find a similar trend where the STGCN graph filters learned in the initial layers capture general human motion, focusing on all the nodes in the S-graph and having class-specific node importance only in a few final layers of the network.

D. EFFECT OF NOISE IN THE DATA

We analyze the robustness of the STGCN network in the presence of noise in the data. In our experiments, we add noise at various peak signal-to-noise ratio (PSNR) levels to a set of actions and compare the label smoothness over the layers concerning the original signal.

A popular approach to incorporate noise into the spatiotemporal data is to add additive white Gaussian noise to the measurement [6]. Fig. 6 shows label smoothness for three actions *drop*, *hop*, and *standing up (from a sitting position)*. It is clear from these examples that the overall performance degraded slightly, while the smoothness of the labels through successive layers of the network is better than the original signal. Specifically for the action *drop*, as we discussed in Section IV-B, the label smoothness degraded in the middle of the network and recovered at the end. However, for the noisy signal, we notice a more stable, non-increasing pattern as in other actions. The accuracy of the STGCN network on this partially noisy dataset is 80.2%. Hence, the network is robust to this additive white Gaussian noise.

E. TRANSFER PERFORMANCE

So far, we see that the first few network layers focus on understanding general human motion, which is needed before learning the specific task. Therefore, the hypothesis is that the network should exhibit similar behavior in the layerwise representations for a similar human activity dataset. To explore the area of adapting the pre-trained STGCN model to a new dataset and analyze the transfer performance of STGCN, we use the new 60 actions (61-120) in NTU-RGB120 [30] dataset. In the rest of the paper, we refer to these new actions 61-120 as NTU-RGB61-120.

We first analyze the label smoothness of the NTU-RGB61-120 dataset on the pre-trained STGCN model trained on the NTU-RGB60 dataset. Fig. 7 (Left) shows the label smoothness over the successive layers of the network. We divided NTU-RGB61-120 into three sets, lower body, upper body, and full body actions, depending on the involvement of the body joints (Table 3) We notice a similar pattern as we observed for NTU-RGB60. There is a big jump in the label

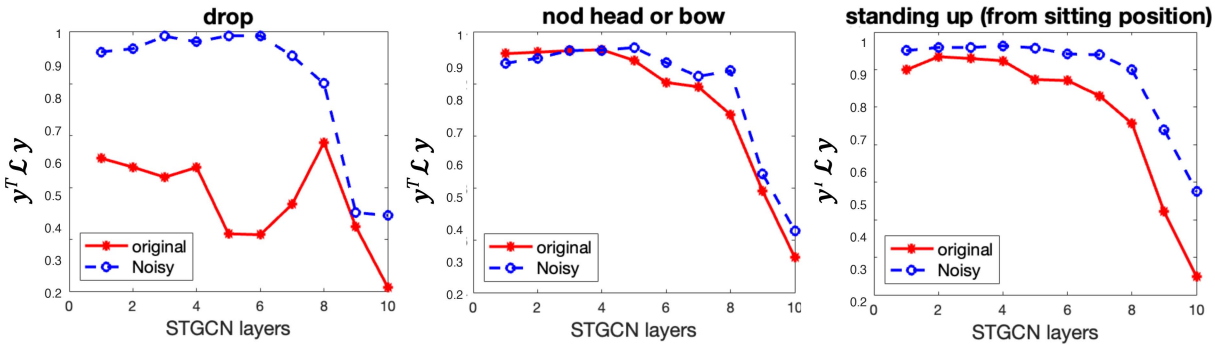

FIGURE 6. Impact of noise added to the input sequence on the label smoothness observed using the corresponding features obtained with the model. The Laplacian quadratic ($\mathbf{y}^\top \mathcal{L}_{out} \mathbf{y}$) decreases as the label smoothness increases. We show the impact of noise on one action per super-class grouping (upper, lower, and full body). We observe that in actions where the Laplacian quadratic had a steady (non-increasing) trend, it remained mostly unaffected by adding noise to the input action sequence. However, actions where the label smoothness decreased before increasing were affected by noise. This implies that the features learned in the early layers for these actions are not robust, and adding noise allows us to see the modified manifold induced in these layers.

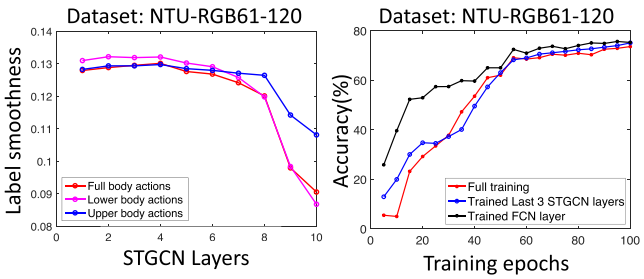

FIGURE 7. **Left:** Label smoothness of unseen action classes (NTU-RGB61-120) using a model trained on NTU-RGB60. We present results averaged over each super-class (Upper, Lower, and Full body). We see that the model embedding allows the features corresponding to new action sequences to be separable. Further, $\mathbf{y}^\top \mathcal{L} \mathbf{y}$ follows a similar non-increasing trend as in the case of the NTU-RGB60 in a much smaller range of scale. This implies that the features learned by the model can be used for the novel classes, and model transfer can be done with simple fine-tuning. **Right:** Classification accuracy on NTU-RGB61-120 test-set using a 10-layer STGCN network. Performance comparison between a model trained from scratch and one obtained with transfer learning by fine-tuning a model trained on NTU-RGB60. We can see the effectiveness of model transfer, which was predicted by our label smoothness analysis.

smoothness after layer 8, while the slope changes slowly before that. Therefore, although the network is not trained on NTU-RGB61-120, it shows similar behavior, proving our hypothesis. The overall accuracy of the network is 9%, which states the need for fine-tuning and it achieves 78% accuracy after fine-tuning.

Fig. 7 (Right) shows the validation accuracy (validation loss in supplementary Fig. 14) of the STGCN network with respect to the training epochs. In the case of transfer learning, we can fine-tune the last few layers depending on the performance or the availability of the data. We consider 3 cases of fine-tuning varying the number of layers, such as 1. training only the FCN layer, 2. training the last 3 STGCN layers, including the FCN layer, and 3. training the whole network. Interestingly, we see that in this case, only fine-tuning the FCN layer (case 1) provides good performance. This means that STGCN captures a good representation of these human motions. If we have a dataset where the actions are very different than the trained dataset, we can fine-tune more layers depending on the availability of the data.

V. CONCLUSION

We present a data-driven approach for understanding STGCN models using windowed-DTW distance-based NNK graphs.

Analyzing the label smoothness of the successive layers on the *NNK Dataset Graph*, we show that the initial layers focus on general human motion, and features for individual action recognition are learned by the model only in the later layers. We also present a comparison between graph construction methods, showing the superiority of the NNK graph over the k-NN graph. To validate our insights from label smoothness, we introduce an L-STG-GradCAM method to visualize the importance of different nodes at each layer for predicting the action. We then present our analysis of label smoothness and its impact on the transfer performance of a trained STGCN model to unseen action classes. Finally, we present an analysis of the robustness of the features at each layer of an STGCN in the presence of input Gaussian noise. We show that the added noise does not affect the label smoothness of several action classes.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, pp. 1097–1105, 2012, vol. 25.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[3] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Jan. 2021.

[4] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger, "Simplifying graph convolutional networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6861–6871.

[5] T.-A. N. Pham, X. Li, G. Cong, and Z. Zhang, "A general graph-based model for recommendation in event-based social networks," in *Proc. IEEE 31st Int. Conf. Data Eng.*, 2015, pp. 567–578.

[6] J.-Y. Kao, A. Ortega, D. Tian, H. Mansour, and A. Vetro, "Graph based skeleton modeling for human activity analysis," in *Proc. IEEE Int. Conf. Image Process.*, 2019, pp. 2025–2029.

[7] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," 2017, *arXiv:1709.04875*.

[8] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018.

[9] B. Ren, M. Liu, R. Ding, and H. Liu, "A survey on 3 D skeleton-based action recognition using learning method," 2020, *arXiv:2002.05907*.

[10] B. Ghorbani, S. Krishnan, and Y. Xiao, "An investigation into neural net optimization via hessian eigenvalue density," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 2232–2241.

[11] S. Arora, S. Du, W. Hu, Z. Li, and R. Wang, "Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 322–332.

[12] S. Gunasekar, J. D. Lee, D. Soudry, and N. Srebro, "Implicit bias of gradient descent on linear convolutional networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018.

[13] R. Baldock, H. Maennel, and B. Neyshabur, "Deep learning through the lens of example difficulty," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, vol. 34, pp. 10876–10889.

[14] S. Shekkizhar and A. Ortega, "Model selection and explainability in neural networks using a polytope interpolation framework," in *Proc. IEEEE 55th Asilomar Conf. Signals, Syst., Comput.*, 2021, pp. 177–181.

[15] R. Cosentino, S. Shekkizhar, S. Avestimehr, M. Soltanolkotabi, and A. Ortega, "The geometry of self-supervised learning models and its impact on transfer learning," 2022, *arXiv:2209.08622*.

[16] Z. Zhang, "Microsoft kinect sensor and its effect," *IEEE Multimedia*, vol. 19, no. 2, pp. 4–10, Feb. 2012.

[17] C. Li, Q. Zhong, D. Xie, and S. Pu, "Co-occurrence feature learning from skeleton data for action recognition and detection with hierarchical aggregation," 2018, *arXiv:1804.06055*.

[18] Y. Yan, J. Xu, B. Ni, W. Zhang, and X. Yang, "Skeleton-aided articulated motion generation," in *Proc. 25th ACM Int. Conf. Multimedia*, 2017, pp. 199–207.

[19] M. Li, S. Chen, X. Chen, Y. Zhang, Y. Wang, and Q. Tian, "Actional-structural graph convolutional networks for skeleton-based action recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 3595–3603.

[20] Z. Liu, H. Zhang, Z. Chen, Z. Wang, and W. Ouyang, "Disentangling and unifying graph convolutions for skeleton-based action recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 143–152.

[21] P. Das and A. Ortega, "Symmetric sub-graph spatio-temporal graph convolution and its application in complex activity recognition," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process*., 2021, pp. 3215–3219.

[22] C. Pan, S. Chen, and A. Ortega, "Spatio-temporal graph scattering transform," 2020, *arXiv:2012.03363*.

[23] D. Bonet, A. Ortega, J. Ruiz-Hidalgo, and S. Shekkizhar, "Channel redundancy and overlap in convolutional neural networks with channel-wise NNK graphs," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process*., 2022, pp. 4328–4332.

[24] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," 2013, *arXiv:1312.6034*.

[25] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 618–626.

[26] P. Das and A. Ortega, "Gradient-weighted class activation mapping for spatio temporal graph convolutional network," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process*., 2022, pp. 4043–4047.

[27] S. Shekkizhar and A. Ortega, "Graph construction from data using non negative kernel regression (NNK graphs)," 2019, *arXiv:1910.09383*.

[28] M. Müller, "Dynamic time warping," *Inf. Retrieval Music Motion*, pp. 69–84, 2007.

[29] 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.*, 2016, pp. 1010–1019.

[30] J. Liu, A. Shahroudy, M. Perez, G. Wang, L.-Y. Duan, and A. C. Kot, "NTU RGB+D 120: A large-scale benchmark for 3D human activity understanding," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 10, pp. 2684–2701, Oct. 2020.

[31] S. Shekkizhar and A. Ortega, "Graph construction from data by non-negative kernel regression," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2020, pp. 3892–3896.

[32] S. Shekkizhar and A. Ortega, "Revisiting local neighborhood methods in machine learning," in *Proc. IEEE Data Sci. Learn. Workshop*, 2021, pp. 1–6.

[33] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with GPUs," *IEEE Trans. Big Data*, vol. 7, no. 3, pp. 535–547, Jul. 2021.

[34] G. Ongie, R. Willett, D. Soudry, and N. Srebro, "A function space view of bounded norm infinite width ReLU nets: The multivariate case," in *Proc. Int. Conf. Learn. Representations*, 2019.

[35] X. Zhu, Z. Ghahramani, and J. D. Lafferty, "Semi-supervised learning using Gaussian fields and harmonic functions," in *Proc. 20th Int. Conf. Mach. Learn.*, 2003, pp. 912–919.

[36] A. Ortega, *Introduction to Graph Signal Processing*. Cambridge, U.K.: Cambridge Univ. Press, 2022.

[37] M. Fazlyab, M. Morari, and G. J. Pappas, "Safety verification and robustness analysis of neural networks via quadratic constraints and semidefinite programming," *IEEE Trans. Autom. Control*, vol. 67, no. 1, pp. 1–15, Jan. 2022.

[38] M. Fazlyab, A. Robey, H. Hassani, M. Morari, and G. Pappas, "Efficient and accurate estimation of lipschitz constants for deep neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019.

[39] V. Gripon, A. Ortega, and B. Girault, "An inside look at deep neural networks using graph signal processing," in *Proc. IEEE Inf. Theory Appl. Workshop*, 2018, pp. 1–9.

[40] C. Lassance, V. Gripon, and A. Ortega, "Representing deep neural networks latent space geometries with graphs," *Algorithms*, vol. 14, no. 2, 2021, Art no. 39.

[41] J. A. Tropp, *Topics in Sparse Approximation*. Austin,TX, USA: Univ. Texas Austin, 2004.

[42] K. Cheng, Y. Zhang, X. He, W. Chen, J. Cheng, and H. Lu, "Skeleton-based action recognition with shift graph convolutional network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 183–192.

[43] W. Kay et al., "The kinetics human action video dataset," 2017, *arXiv:1705.06950*.

[44] D. Osokin, "Real-time 2D multi-person pose estimation on CPU: Lightweight OpenPose," in *Proc. 8th Int. Conf. Pattern Recognit. Appl. Methods. SCITEPRESS-Sci. Technol. Pub.*, 2019.