Ants: Attacking Spatial Temporal Graph Learning Networks Structurally

Ran Ran*, Qi Liu†, Nuo Xu‡, Ning Sui*, and Wujie Wen*
*North Carolina State University, †Amazon, ‡Lehigh University
*{rran, nsui, wwen2}@ncsu.edu, †liuqim@amazon.com, ‡nux219@lehigh.edu

2024 IEEE 48th Annual Computers, Software, and Applications Conference (COMPSAC) | 979-8-3503-7696-8/24/$31.00 ©2024 IEEE | DOI: 10.1109/COMPSAC61105.2024.00139

Abstract—Spatial-temporal graph data widely appear in real-world applications like traffic flow forecasting and skeleton-based action recognition. To effectively handle such data, spatial-temporal graph learning neural networks (STGNNs) have emerged as a promising solution and outperform traditional deep learning in these tasks. However, existing studies mainly focus on the performance aspect but lack a systematic study on their robustness when spatial-temporal graph structure varies. In this paper, we aim to fill this gap by systematically investigating the structure vulnerabilities of a variety of popular STGNNs in practical tasks. To this end, a spatial-temporal graph attack framework, namely *Ants*, which consists of gradient-sign guided sensitive time-slice selection followed by quantization-based adversarial edge searching, is proposed to craft minimized adversarial perturbations on discrete spatial-temporal graph data in both spatial and temporal dimensions. Extensive experiments on traffic flow forecasting and human action recognition tasks with six STGNNs and four datasets well demonstrate the effectiveness and stealthiness of our *Ants*. While prior works do not consider attacks on spatial-temporal graphs, we compare our attack with state-of-the-art attacks on graphical models extended to the STGNN setup, and the performance of *Ants* demonstrates that, for STGNNs, the structure vulnerability in the context of spatial and temporal dynamics needs to be considered carefully.

***Index Terms*—Adversarial Attack, Spatial-Temporal, Graph Neural Network**

I. Introduction

Spatial-temporal graph, as a special graph data with spatial-temporal semantics, plays an increasingly important role in many practical applications, including human action recognition and traffic flow forecasting. Unsurprisingly, deep learning on spatial-temporal graph data has seen increased research exploration recently. In particular, a growing number of spatial-temporal graph neural networks (STGNNs) have demonstrated competitive performance in a range of spatial-temporal graph applications [1], [2].

However, the robustness study of STGNNs, which is of paramount importance to practical tasks in addition to performance, is still in its infancy. While there exist some attempts [3], [4] , e.g., graph adversarial examples, they mainly focus on adding perturbations on the properties or features of graph node (e.g., the joint position of the skeleton in human action recognition), leaving the important graph structure-based attacks against STGNNs, largely unexplored. Indeed, it is essential to understand how changing graph structure impacts the performance of STGNNs in the context of spatial and temporal dynamics. As we shall show in Sec. V, for graph classification tasks like human action recognition, by just altering a very small amount of edges without touching node features of spatial-temporal graph data, STGNN would precisely mispredict the action that may impact the critical real-time response of self-driving cars.

In response to this, this work aims to explore a graph-structure adversarial attack dedicated to spatial-temporal graph data. Crafting new adversarial graph attacks under the context of spatial and temporal dynamics faces the following challenges: **1) Two-dimensional optimization complexity over space and time.** Unlike continuous data such as image and video, the spatial-temporal graph structure is discrete. Therefore, we need to design efficient optimization algorithms to find adversarial examples across space and time in the discrete domain. There exist prior works which [5], [6] focus on adding adversarial perturbation on the discrete "spatial-only" graph for graph neural networks in graph-based tasks. However, we found that directly using existing solutions to search adversarial perturbations on the spatial-temporal graph cannot ensure a high attack success rate given the small edge perturbation budget (see Table VII). This prompts the need for optimized solutions dedicated to the spatial-temporal graph. **2) Attack stealthiness requirements across space and time.** Different from existing "spatial-only" graph-based attacks, STGNN attacks should be stealthy in spatial and temporal dimensions while achieving effective attacks. However, *the scale of spatial graph per time-slice* in practical spatial-temporal tasks is usually smaller than traditional "spatial-only" graph applications because of physical constraints such as distance or structure, e.g., hundreds of nodes/edges in traffic network at a certain time-slice [2] v.s. tens of thousands of nodes/edges in social networks. This imposes a tighter margin to hide attacks in the spatial dimension. Further, the attackers need to ensure that distorted time-slices are limited to achieve stealthiness in the temporal dimension.

To this end, we propose a spatial-temporal graph adversarial attack framework, namely structural spatial-temporal attack (***Ants***) against STGNNs. **First**, to efficiently search for adversarial examples on the complex spatial-temporal graph and achieve a stealthy attack in the temporal dimension, we develop a gradient sign-based time-slices selection algorithm to quickly identify a few most sensitive time-slices from the temporal dimension for generating adversarial examples. **Second**, we propose adaptive quantization optimization to search for a spatial-temporal graph adversarial example in the discrete domain. We evaluate our attack on two representative applications: traffic flow forecasting and human action recognition under six types of STGNNs and four datasets considering both white-box and black-box settings. Experiments show that *Ants* achieves competitive attack effectiveness and stealthiness in both temporal and spatial dimensions. For example, in the white-box setting, *Ants* achieves 94.5% average success rate by changing less than 1% edges on the human action recognition task.

II. Background

Spatial-Temporal Graph Neural Networks (STGNNs). STGNNs aim to learn the pattern from spatial-temporal graphs by capturing the graph's spatial and temporal dependencies. The studies of STGNNs are mainly active in two fields: traffic flow forecasting and human action recognition. **1) STGNNs in traffic flow forecasting.** Guo et al. [2] propose a spatial-temporal graph

2836-3795/24/$31.00 ©2024 IEEE
DOI 10.1109/COMPSAC61105.2024.00139

convolution that utilizes graph convolution to capture spatial patterns and a standard convolution to extract the temporal features.

Moreover, the attention mechanism is adopted to adaptively capture dynamic spatial and temporal correlation of traffic data. As a result, GNN without attention mechanism is named *Multi-component Spatial-temporal Graph Convolution Network (MSTGCN)*, while the one with attention mechanism is called *Attention based Spatial-temporal Graph Convolution Network (ASTGCN)*. Besides, *Spatial-Temporal Synchronous Graph Convolutional Network (STSGCN)* [7] uses a spatial-temporal synchronous modeling mechanism to capture the localized spatial-temporal correlations instead of using two separate components like ASTGCN. **2) STGNNs in human action recognition.** Unlike the traffic network, the human skeletal structure as a graph is relatively simple. However, there is a large number of human action categories. Thereby, most studies focus on abstracting hidden graph features from the physical graph structure before feeding it to GNN for training. *Spatial Temporal Graph Convolutional Network (ST-GCN)* [8] introduces three human-defined graph partition strategies [1] to obtain abstracted graph from skeletal structure. *Actional-Structural Graph Convolutional Network (AS-GCN)* [8] uses a trainable auto-encoder to capture fine-grained graph features from both skeletal structure and action, and then combines the high-order polynomial of the skeletal graph as final graph input. Liu et al. [9] propose a powerful feature extractor named *MS-G3D* that utilizes a multi-scale aggregation technique to disentangle graph features and a unified spatial-temporal graph convolution to capture cross-space-time joint dependencies.

Adversarial Attack can mislead deep learning models by adding unnoticeable perturbations to input. There exist adversarial examples against various types of data, including images [10], video [11], and graph [5]. We focus on studying graph structure-based adversarial attacks in this work. Most graph adversarial studies focus on "spatial-only" graphs and evaluate attacks in traditional graph applications such as citation networks. For example, Meta-attack [5] applies meta-gradient to construct a score function to obtain the most sensitive edge. EpoAtk [6] overcomes the limitation of the meta-attack that always uses the maximal gradient by introducing a fine-grained searching process with the generation, evaluation, and recombination. It results in a better attack performance. Only a few works [3], [4], [12] have recently started to study adversarial attacks on the spatial-temporal graph in STGNNs. However, their focus is perturbing the graph node/vertex instead of the graph structure. For example, [3] perturbs the locations of body joints (i.e., node properties on the graph) to generate adversarial examples and utilizes Generative Adversarial Network (GAN) to make generated examples more natural. A recent work [12] focuses on selectively attacking the most sensitive graph node's features based on the magnitude of the gradient to craft effective attacks, and it is limited to traffic flow estimation task only. In general, *our work differs in two aspects from them: 1) graph-structure attacks by modifying graph edges in spatial-temporal domains v.s. node property attacks by perturbing the properties of graph nodes; 2) more general spatial-temporal graph applications (e.g., node property prediction in traffic flow forecasting), in addition to human action recognition for graph classification.*

III. Attack Formulation

We formulate our proposed spatial-temporal graph attacks by the following representative STGNN tasks: 1) node property prediction in time series; and 2) graph classification. For both tasks, the spatial-temporal graph input data is denoted as $G{=}(A,X){=}(\{A_t\}_{t=1}^{M},\{X_t\}_{t=1}^{M})$, where $A{\in}\{0,1\}^{M\times N\times N}$ and $X{\in}\mathbb{R}^{M\times N\times F}$ are the discrete adjacent matrix and the node properties, respectively. (M, N, F) denote the number of time-slices, the number of nodes, and the number of properties per node, respectively. **For Task 1–node properties prediction**, the goal is to predict the node properties of the next T_p time-slices. Assuming ground truth node properties are $Y = \{y_t\}_{t=M+1}^{M+T_P}$, the parameter θ of a STGNN model f can be learned by minimizing the loss function $\mathcal{L}$ defined by Mean Square Error (MSE), $\mathcal{L}{=}\frac{1}{T_p\cdot N}\sum_{i=M+1}^{M+T_p}||f_\theta(G)_i - y_i||_2^2$, in which $f_\theta(G)_i$ is the predicted node property at time-slice t. To evaluate attack performance for Task 1, we use **Mean Absolute Error (MAE) as the metric** to measure how close prediction results and ground truth are: $\text{MAE}_{(f_\theta(G),Y)} = \frac{1}{T_p}\sum_{t=M+1}^{M+T_p}|f_\theta(G)_t - Y_t|$. We define $f_\theta(A,X){\neq}Y$, when the $\text{MAE}_{(f_\theta(G),Y)}{>}0$. **For Task 2–graph classification**, each graph $G{=}(A,X)$ is assigned a one-hot label $Y{=}\{y_1,y_2,...,y_C\}$, where C is the number of classes. Training a STGNN model can be achieved by minimizing the cross entropy loss: $\mathcal{L}{=}-\frac{1}{C}\sum_{i=1}^{C} y_i log(f_\theta(G)_i)$, where y_i is the binary indicator for class i, $f_\theta(G)_i$ is the predicted probability of class i. To evaluate attack performance for Task 2, we use **attack success rate as the metric**, where it is defined as $f_\theta(A, X) \neq Y$, when the predicted label does not match with ground truth. We take traffic flow forecasting and human action recognition as example of tasks 1 and 2 to demonstrate the proposed attacks.

A. Threat Model

Attacker's goal. The attacker aims to insert/remove edges of graph to change A to A' to achieve a successful attack. We formulate it as an optimization problem subject to **both temporal and spatial constraints** simultaneously:

$$\begin{aligned} &\arg\min_{A'}\{\mathcal{L}_{AE}(A', X)\} \\ &\text{s.t. } f_\theta(A', X) \neq Y \\ &\sum_{t=1}^{M}\Delta A_t \leq \lfloor \sigma M \rfloor, \sigma \in (0, 1] \\ &\|A_t - A'_t\|_0 \leq \lfloor \eta\frac{N^2}{2}\rfloor, \eta \in (0, 1),\ \forall t \in [1 : M] \end{aligned} \tag{1}$$

where A' is the only variable in the above optimization problem, $\mathcal{L}_{AE}$ is the attack loss function. For node property prediction, we define $\mathcal{L}_{AE}$ as the negative loss for training, i.e., $\mathcal{L}_{AE}{=}{-}\mathcal{L}$. For graph classification, we adopt the C&W loss [13] used in adversarial attacks to image models: $\mathcal{L}_{AE}{=}f(G')_y - \max_{i\neq y} f(G')_i$. Note that we do not need future time-slices' ground truth labels. Instead, we adopt the trained model (before performing the attack) to predict the node/graph labels for future graphs based on the current time stamp, and then minimize $\mathcal{L}_{AE}$ to make the model's prediction deviate y. ΔA_t indicates whether a graph A_t is changed or not during the attack, i.e., $\Delta A_t{=}0, \text{if } A_t{=}A'_t$; and 1, otherwise. The second condition is thus a **temporal constraint** that limits the fraction of perturbed time-slice graphs to be no large than σ. The third condition is a **spatial constraint** that restricts the maximum fraction of the perturbed edges within a time-slice to be η.

Attacker's knowledge. STGNN mainly includes three components: parameter θ, training data (i.e., graph $G{=}(A,X)$), model structure and type. According to the level of maintained knowledge, we divide the attack into *white-box attack* and *restricted black-box attack*. For the former, the attacker has full knowledge of the target model and training data. For the latter, the attacker does not know parameters but has access to training data and model type, and can train a substitute model to ensure the transferability of the attack [14] on the target model.

Attacker's capability. Similar to existing adversarial graph attacks [5], we assume the attacker is capable of modifying the graph structure (i.e., adding or removing edges) in the spatial-temporal

Algorithm 1: Overview of *Ants*

Data: STGNN $f(A, X)$, spatial-temporal graph data G, number of nodes N, edges constraint η, time-slices constraint σ, maximum iteration number n, the initial/increased number of selected time-slices T_0/T_k, step size of time-slices updating s.
Result: adversarial graph data $G' = (A', X)$

1 success, i, $T \leftarrow$ False, 1, T_0
2 $m_h \leftarrow \mathbf{1}_{T\times N\times N} - I$ // initialize mask for quantization optimization
3 $A_{sub} \leftarrow$ *TimeSliceSelection*(A, T_0)
4 $A', A^{'}_{sub}, \leftarrow A, A_{sub}$
5 **while** *!success* & $\|A_t - A^{'}_t\|_0 \leq \lfloor \eta \frac{N^2}{2} \rfloor$ & $i < n$ **do**
6 **if** *i % s == 0* & $T < \lfloor \sigma M \rfloor$ **then**
7 $A^{'}_{sub} \leftarrow A^{'}_{sub} \cup$ *TimeSliceSelection*($A \backslash A_{sub}$, T_k)
8 $T \leftarrow T + T_k$
9 $A^{'}_{sub}, m_h \leftarrow$ *QuanBasedAdvGraphCreate*($A^{'}_{sub}, A, m_h$)
10 $A' \leftarrow$ construct entire adversarial graph using $A^{'}_{sub}$ and A
11 **if** $f(A^{'}, X) \neq f(A, X)$ **then** success = True
12 i $\leftarrow$ i+1
13 **return** $G' = (A', X)$

graphs. For example, an attacker can hijack the communication channel between distributed edge sensors which generate graph data, and the remote cloud server that hosts STGNN models for efficient inference. Then the attacker could hack the graph structure by carefully altering graph edges (a.k.a structure perturbation). Here we assume the node features are not hacked since the graph structure perturbations have already led to a high attack success rate.

IV. The Design Overview of *Ants*

Directly solving Eq.1 is computationally intractable in that it is a discrete optimization problem. Here we develop a spatial-temporal interactive framework, namely adaptive quantization-based spatial-temporal attack **(*Ants*)**, to solve Eq.1 in a feasible way. *Ants* includes two main modules: **(M1)** sensitive time-slice selection and **(M2)** quantization-based spatial-temporal adversarial graphs generation. In particular, **M1** is guided by a fast gradient sign-based method to quickly identify the most sensitive time-slices of spatial-temporal graph data in a coarse-grained manner. **M2** further applies adaptive quantization-based optimization to time-slices selected by **M1**. By iterating **M1** and **M2**, highly evasive and effective spatial-temporal adversarial graphs can be generated efficiently.

Algorithm 1 shows an overview of *Ants*. To reduce the perturbations in both spatial and temporal dimensions and make generated adversarial graphs more effective, we will execute two modules (i.e., *TimeSliceSelection* and *QuanBasedAdvGraphCreate*) iteratively. Specifically, we conduct *TimeSliceSelection* to search T_0 most sensitive time-slices and construct a sub-graph A_{sub} (line 3). Then we perform *QuanBasedAdvGraphCreate* to generate (or update) the adversarial sub-graph and validate the reconstructed adversarial graph's attack effectiveness (line 9-11). If the attack cannot succeed in a limited number of iterations, we will add more time-slices (i.e., T_k) into the current sub-graph by calling *TimeSliceSelection* every few iterations (controlled by step size s) until the total number of selected time-slices (i.e., T) reaches the constraint $\lfloor \sigma M \rfloor$ (line 6-8). Finally, we constantly update the adversarial graph until the attack succeeds or the constraints are violated. We present the design details of **M1** and **M2** in Section A and B, respectively. We experimentally validate and report the time cost in Table VII.

A. M1: Gradient Sign Time-slice Selection

Different time-slices on spatial-temporal graphs could have different sensitivities to an attack. Hence, to generate an effective adversarial graph with few edge perturbations in the temporal domain, it is essential to measure the importance of each time-slice and select the relatively sensitive time-slices.

Algorithm 2: Adaptive Quantization-based Adversarial Graph Generation

Data: adversarial sub-graph $A^{'}_{sub}$, clean graph A, mask m_h, loss function $\mathcal{L}_{AE}$, learning rate lr, clipping threshold h, quantization coefficient sequence $q_s = [\tau_0, \tau_0 + g, \tau_0 + 2 \cdot g, \ldots, \tau_n]$.
Result: adversarial graph structure A' and mask m_h.

1 $A^{'}_{sub} \leftarrow A^{'}_{sub} - lr \cdot \nabla_{A^{'}_{sub}} \mathcal{L}_{AE} \odot m_h$;
2 $A^{'}_{sub}, m_h \leftarrow$ clip $A^{'}_{sub}$ to $(-h, 1 + h)$ and update m_h;
3 $A_{sub} \leftarrow A_{sub} \odot (\mathbf{1} - m_h) + A^{'}_{sub} \odot m_h$;
4 $\tau \leftarrow$ obtain optimal quantization coefficient τ from q_s ;
5 $A^{'}_{sub} \leftarrow Q(A_{sub}, A^{'}_{sub}, \tau)$;
6 **return** $A^{'}_{sub}, m_h$

To find T most vulnerable time-slices, we design a coarse-grained strategy that quickly evaluates the impact of (a few) edge modifications on the loss function per time-slice. This consists of two steps: **i) intra-time-slice search (for n_e most sensitive edges); ii) cross-time-slice search (for T most vulnerable time-slices). For Step i**, we identify the most sensitive edges per time-slice by ranking the absolute value of the gradient $\nabla_{A_t} \mathcal{L}_{AE}$ within that time-slice $G_t = \{A_t, X_t\}$, which can be expressed as $e_t = TOP_{n_e}|\nabla_{A_t} \mathcal{L}_{AE}(f(X,A))|, t = 1, .., M$, where TOP function returns the index of n_e sensitive edges at time-slice t. To determine how to change selected sensitive edges in each time-slice independently for coarse-grained attack evaluation, we adapt the fast gradient sign method (FGSM) [10] for continuous data (e.g., image) to discrete graph data. Specifically, given a binary graph data A_t and $sign(\nabla_{A_t} \mathcal{L}_{AE})$, if $A_{t,ij}=1$ and $sign(\nabla_{A_{t,ij}} \mathcal{L}_{AE})=-1$ or $A_{t,ij}=0$ and $sign(\nabla_{A_{t,ij}} \mathcal{L}_{AE})=1$, then $A_{t,ij}$ changes as "$1 \rightarrow 0$" or "$0 \rightarrow 1$". Then, we define a mask m_t to filter the top e_t sensitive edges as below:

$$m_t = A_t \oplus \left(\frac{sign(\nabla_{A_{e_t}} \mathcal{L}_{AE}) + 1}{2} \right) \tag{2}$$

where $\oplus$ is the "Exclusive OR" logic operator. Then the per time-slice graph with the perturbation becomes $A_t + sign(\nabla_{A_{e_t}} \mathcal{L}_{AE}) \odot m_t$, where $\odot$ is the Hadamard Product. **For Step ii**, we generate an attack loss profile set $(\mathcal{L}^1, \mathcal{L}^2, \ldots, \mathcal{L}^M)$ for all M time-slice graphs after perturbation based on **Step i**. Finally, time-slices corresponding to the T largest losses $(TOP_T\{\mathcal{L}^t\}_{t=1}^M)$, will be selected as the candidates sub-graphs for optimized spatial-temporal adversarial graph generation (**M2**).

B. M2: Adversarial Graph Generation

Once sensitive time-slices are selected, our next step is to generate optimized spatial-temporal adversarial graphs with least edge perturbations. To this end, we propose an adaptive quantization-based optimization to effectively and efficiently generate spatial-temporal adversarial graphs.

Given that all elements in a graph A should be either 0 or 1, we propose to quantize an adversarial graph A'. One naive solution is as follows:

$$Q(A')_{ij} = \begin{cases} 1 & \text{if } A'_{ij} \geq \frac{1}{2} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

However, this can bring a large fixed quantization error (e.g. $1/2$) at all iterations and hence hurt the optimization process significantly. To tackle this problem, we propose a fine-grained adaptive quantization function as below:

$$Q(A', \tau)_{ij} = \begin{cases} 1 & \text{if } A'_{ij} \geq A_{ij} \cdot \tau + (1 - A_{ij}) \cdot (1 - \tau) \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

TABLE I
RESULTS OF APPLYING THE ROUND FUNCTION AND ADAPTIVE QUANTIZATION WITH DIFFERENT GAPS g (AQ(g)).

Quantization function	Round function	AQ(0.2)	AQ(0.1)	AQ(0.05)
Success rate	74%	90.5%	93.2%	93.6%
# changed edges	226	168	157	154
Time cost	204s	220s	284s	527s

TABLE II
RESULTS OF VARYING INITIALLY SELECTED TIME-SLICES IN *Ants*.

T_0	5%	10%	15%	20%	25%	30%
Success rate (%)	80.6%	88.5%	93.6%	93.8%	93.7%	93.7%
# changed edges	226	183	155	152	158	158
# changed time-slices	33	35	38	42	50	54
Time cost (s)	224s	285s	326s	389s	475s	572s

where $\tau \in (0,1)$ is to adjust the quantization threshold by considering value changes of A' at each iteration, thereby offering a varied quantization error–τ during the optimization. Since the goal is to find "adversarial edges", we could start with a small τ which incurs a small quantization error, and then gradually increase τ if the attacker's goal is hard to achieve. To refine the quantization process, we adjust the quantization threshold at every few iterations: $[\tau_0, \tau_0+g, \tau_0+2\cdot g, \ldots, \tau_n]$, where τ_0 (τ_n) is the initial (final) value, and g is the quantization gap. Then we approximately replace the gradient $\nabla_A \mathcal{L}_{AE}$ as $\nabla_{Q(A',\tau)} \mathcal{L}_{AE}$ in each iteration. Accordingly, the gradient update can be expressed as $\left(\nabla_{Q(A',\tau)} \mathcal{L}_{AE}\right)_{i+1} = lr \cdot \left(\nabla_{Q(A',\tau)} \mathcal{L}_{AE}\right)_i \odot m_h$. Here lr is the learning rate, and m_h is a mask. We design a heuristic clipping strategy to update the mask m_h to reduce the search space of potential adversarial edges automatically, which can be represented as:

$$m_{h,ij} = \begin{cases} 0 & \text{if } (A'_{ij} - \nabla_{Q(A',\tau)_{ij}} \mathcal{L}_{AE} \geq h+1 \ \& \ A_{ij} = 1) \\ & \text{or } (A'_{ij} - \nabla_{Q(A',\tau)_{ij}} \mathcal{L}_{AE}) < -h \ \& \ A_{ij} = 0) \\ 1 & \text{otherwise} \end{cases} \quad (5)$$

where $h \in [0,1)$ is a heuristic clipping threshold. Once $A'_{t,ij}$ exceeds the threshold (h), we will stop updating $A'_{t,ij}$ and reset it to the original (i.e., $A_{t,ij}$). The details are shown in Algorithm 2.

C. Time and Space Complexity Analysis

Time complexity. We analyze the time complexity of our attack from spatial and temporal dimensions, respectively. In the temporal dimension, our algorithm needs to traverse M time-slices to select n_e most vulnerable time-slices, and then to traverse selected n_e time-slices to generate an adversarial graph, leading to time complexity of $\mathcal{O}(M+n_e)$. In spatial dimension, we need to compute gradients of each element in the adjacency matrix $A \in \{0,1\}^{N \times N}$ for each selected time-slice, which leads to time complexity of $\mathcal{O}(N^2)$. Thus, in the worst case, the total time complexity of our attack is $\mathcal{O}(n \cdot (M+n_e \cdot N^2))$ (n_e is a constant), where n is the maximum number of iterations. Especially, because of the elaborate design in spatial and temporal domains, the time complexity is reduced from the standard $\mathcal{O}(n \cdot M \cdot N^2)$ (e.g., adversarial graph from a 3D graph directly) to $\mathcal{O}(n \cdot N^2)$).

Space complexity. Since we need to store gradients in the adjacency matrix for M time-slices, the space complexity should be $\mathcal{O}(M \cdot N^2)$. We experimentally validate and evaluate the time cost against SOTA solutions in Table VII.

That is the reason that the time cost of our attack algorithm is much lower than existing adversarial graph attacks (e.g., our attack 326s v.s. EpoAtk 12963s on AS-GCN, see Table 5 on page 6).

V. EVALUATION

A. Experimental Setup

All simulations are conducted in a workstation with one AMD Ryzen Threadripper PRO 3975WX 32-Cores Processor and two NVIDIA GeForce RTX 3090 GPUs.

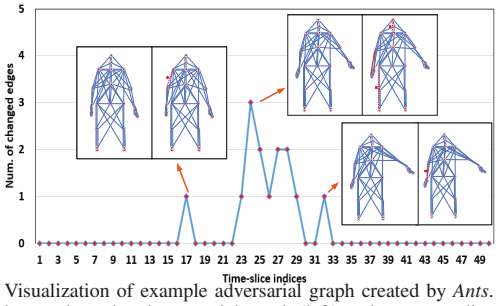

Fig. 1. Visualization of example adversarial graph created by *Ants*. Three double boxes show the clean spatial graph (left) and corresponding adversarial graph (right) in 17-, 24-, 32-th time-slice. The red dotted (solid) line means removing (adding) edge.

Datasets. For the human action recognition task, we evaluate our attack on two datasets: *NTU* [15] and *Kinetics* [16]. *NTU* is the largest dataset with 3D joint annotations for human action recognition, which contains 56K action clips and covers 60 action classes. *Kinetics* is the largest unconstrained action recognition dataset with around 300K video clips collected from YouTube with 400 action classes. For the traffic flow forecasting task, we use two highway traffic datasets: *PeMSD8* and *PeMSD4* collected by the Caltrans Performance Measurement System (PeMS) [17]. *PeMSD8* is traffic data including 1979 detectors on 8 roads in San Bernardino from 07/2016-08/2016. *PeMSD4* involves traffic data in San Francisco Bay area, containing 3848 detectors on 29 roads from 01/2018-02/2018.

Target models. We evaluate *Ants* across 6 different STGNN architectures. ST-GCN [1], AS-GCN [8], and MS-G3D [9] are evaluated on human action recognition, while MSTGCN, ASTGCN [2], and STSGCN [7] are used in traffic flow forecasting. For evaluation generality, these selected STGNNs consist of graphs with static and dynamic structures. For human action recognition, the baseline accuracy of well-trained ST-GCN, AS-GCN, and MS-G3D is 88.1% (52.8%), 93.9% (56.4%), and 96.1% (60.7%) on NTU (Kinetics) dataset, respectively (Top-5 accuracy for Kinetics). For traffic flow forecasting, the baseline MAE of MSTGCN, ASTGCN, and STSGCN is 19.04 (23.85), 18.78 (23.14), and 17.25 (21.28) on PeMSD8 (PeMSD4), respectively. The model accuracy of all baselines **(without attack)** are consistent with that of original papers.

Attack configuration. We derive two attack variants: **1)** ***Ants*** $^-$: an attack that replaces proposed gradient sign time-slice selection (M1) with a simple time-slice selection method adopted in video-based adversarial attack for RNN+CNN [18], i.e., choose the first few time-slices in M2. **2)** ***Ants***: an attack with all proposed techniques. In Algorithm 1, we set the initial number and increment number of time-slices as $T_0 = 15\%M$ and $T_k = 5\%M$ (i.e., $T_0 = 45/37, T_k = 15/12$ for action/traffic datasets). We enforce spatial-temporal constraints on perturbations: $\eta = 0.01$ (change 1% edges in each time-slice at most), and $\sigma = 0.3$ (distort 30% time-slices at most). The first 1000 test data of human action dataset and all test data of traffic datasets are used to generate corresponding adversarial examples, respectively.

Metrics. We evaluate the effectiveness and stealthiness of the attack. The attack success rate and MAE variation are used to measure attack effectiveness in human action recognition and traffic flow forecasting, respectively. The number of changed edges (in the entire spatial-temporal graph) and attacked time-slices are used to validate the attack stealthiness of both tasks.

B. Ablation Study

To demonstrate the generalization of our attack, we analyze key hyper-parameters based on the AS-GCN and NTU dataset, and then

TABLE III
THE ATTACK PERFORMANCE OF OUR ATTACKS ON HUMAN ACTION RECOGNITION AND TRAFFIC FLOW FORECASTING (WHITE-BOX).

Attack success rate (human action recognition)								MAE variation (traffic flow forecasting)						
Model	ST-GCN		AS-GCN		MS-G3D		AVG	MSTGCN		ASTGCN		STSGCN		AVG
Dataset	NTU	Kinetics	NTU	Kinetics	NTU	Kinetics		PeMSD8	PeMSD4	PeMSD8	PeMSD4	PeMSD8	PeMSD4	
Ants $^-$	89.1%	85.3%	78.2%	76.4%	75.4%	78.2%	80.4%	+17.39	+15.97	+18.98	+17.54	+18.93	+18.24	+17.93
Ants	97.5%	92.8%	93.6%	92.1%	94.2%	96.7%	94.5%	+19.23	+18.11	+21.59	+20.23	+23.17	+22.82	+20.85

TABLE IV
THE ATTACK SUCCESS RATE AND MAE VARIATION OF OUR *Ants* ON NTU(VIEW) AND PEMSD8 IN BLACK-BOX SETTINGS.

	ST-GCN (sub)	AS-GCN (sub)	MS-G3D (sub)		MSTGCN (sub)	ASTGCN (sub)	STSGCN (sub)
ST-GCN	79.5%	n/a	n/a	MSTGCN	+15.52	+12.44	n/a
AS-GCN	n/a	72.3%	n/a	ASTGCN	+14.73	+18.01	n/a
MS-G3D	n/a	n/a	73.4%	STSGCN	n/a	n/a	+19.12

TABLE V
RESULTS OF ATTACKING RANDOM TIME-SLICES USING OUR *Ants* IN ST-GCN ON HUMAN ACTION RECOGNITION.

# of attacked time-slices	5%	10%	20%	30%
Attack success rate	80.6%	86.5%	94.3%	96.2%
# of changed edges	12 (0.08%)	14 (0.09%)	18 (0.12%)	21 (0.14%)

directly adopt the values of these hyper-parameters for evaluations on other models and datasets.

Optimized quantization coefficient sequence. Table I shows the impact of using different quantization gap g on attack effectiveness and efficiency. Compared with the round function, our three AQ(g) settings are better in terms of the number of changed edges and success rate. This is because the proposed adaptive quantization function smoothens the minimization process of the loss function, so "better" edges with a fewer number can be identified to achieve attack effectiveness and stealthiness simultaneously. Among the three AQ(g), a smaller g indicates a smoother quantization optimization process and therefore could increase attack effectiveness and reduce the number of changed edges. However, this would increase the time cost. In practice, we found that a very small g does not significantly benefit attack effectiveness and stealthiness despite the increased optimization time, especially for more complex traffic datasets. Therefore, we adopt a hybrid strategy: using a large g (i.e., 0.1) to accelerate the optimization process early on and subsequently reducing g (i.e., 0.05) for a smoother search towards the end of the optimization. The final quantization coefficient sequence is as follows: $q_s = [0.1, 0.2, \ldots, 0.5, 0.55, 0.6, 0.65, \ldots, 0.9]$.

Number of initially selected time-slices. An appropriate number of initially selected vulnerable time-slices (T_0) can promote attack performance and efficiency. In Table II, we observe that too few selected time-slices (e.g., $T_0 = 5\%$ and $10\%M$) lead to a lower success rate and need more changed edges due to limited search scope for "adversarial edges" at the initial stage. However, too large T_0 (e.g., $T_0 = 25\%$ and $30\%M$) could change more time-slices and incur higher time costs. To ensure a high success rate and fewer edge changes, we select $T_0 = 15\%M$ for all STGNNs and datasets.

C. Result and Analysis

White-box attack. Table III reports our evaluation results on two tasks under the white-box attack. For human action recognition, the attack variant–*Ants* $^-$ performs worse than our *Ants* (80.4% vs. 94.5% on average). This is because our gradient sign-based time-slice selection can always identify the most vulnerable time-slices to improve attack performance, while simply picking the first few time-slices as attack targets does not work well in STGNNs, especially for more advanced models (e.g., MS-G3D). We can observe a similar trend in traffic flow forecasting tasks.

Ants can be also easily adapted to the specific time-slices attack and targeted attack. For the specific time-slices attack, we can replace M1 stage with a random time-slices selection method (i.e., selected time-slices randomly), and then construct adversarial examples in the randomly selected time-slices in M2 stage. As Table V shows, even if we only select 5% time-slices randomly, the attack success rate still can reach 80.6%. For the targeted attack, we can slightly change the loss function. Take the traffic flow prediction task as an example, we can use $\mathcal{L}_{AE=\mathcal{L}(o,t)}$ where o is the output of the model, and t is the targeted output expected by the attacker. In our experiment, we randomly select one of the node features as the target (e.g., traffic speed) and set it as the targeted value (e.g., overestimate a specific feature by enlarging its original value by 50%). As Table VI shows, with just $< 1\%$ edge changes, the MAE variation in the targeted feature can be more than 10. For example, for STSGCN, when we only change 0.3% edges, the MAE variation in the targeted feature can be +10.23 (the original MAE without attack is 19.76).

TABLE VI
RESULTS OF TARGETED ATTACK USING OUR *Ants* ON TRAFFIC FLOW FORECASTING.

Model	MAE variation	# of changed edges	# of changed time-slices
MSTGCN	+10.29	0.43%	7.9%
ASTGCN	+10.32	0.41%	7.5%
STSGCN	+10.23	0.3%	8.3%

Visualization for adversarial examples. Fig. 1 shows the first 50 out of 300 time-slices of adversarial graph generated by our *Ants* based on AS-GCN and NTU dataset. It misleads the prediction from "point something" to "taking a selfie" by only a few edges changes. We observe that most time-slices incur no edge changes because our attack only pinpoints the most sensitive time-slices to improve the imperceptibility (in the temporal domain). In this case, only 29 out of 300 time-slices get more than one edge change. On the other hand, for time-slices with edge changes, the edge modifications in 17-, 24-, 32-th time-slice are unnoticeable (in the spatial domain). This is because our attack can find the least number of edge changes. *Thus our attack can achieve stealthiness in both temporal and spatial domains*.

Restricted black-box attack. We adopt a common substitute model-based strategy to achieve the restricted black-box attack by transferability [14]. It uses a local substitute model to craft adversarial examples (the attacker knows everything about the substitute model, white-box setting) to mislead the targeted model (black-box). We train the corresponding substitute model with a reduced structure compared with the targeted model for each type of model, to achieve acceptable performance. Note that, because the graph structure across some STGNNs (e.g., ST-GCN, AS-GCN, MS-G3D) could be different for the same raw graph data, they cannot attack each other. As Table IV shows, the attack success rate or MAE variation of our attack is still high for human action recognition or traffic flow forecasting, indicating a high attack transferability on various STGNNs.

D. Compare with Solutions Augmented from Existing Spatial-only Graph Attacks

To better evaluate our spatial-temporal graph attack–*Ants*, we also compare it with four baselines augmented from the recently developed representative "spatial-only" graph attacks against classic 2D graph tasks: PGD projection [19], GradArgmax [20], meta-attack [5]

TABLE VII
COMPARISON BETWEEN OUR *Ants* AND THREE BASELINES ON NTU DATASET (FIRST 100 TEST DATA).

Metric	Attack success rate			Num. of changed edges			Num. of changed time-slices			Time cost (s)		
Model	ST-GCN	AS-GCN	MS-G3D	ST-GCN	AS-GCN	MS-G3D	ST-GCN	AS-GCN	MS-G3D	ST-GCN	AS-GCN	MS-G3D
PGD proj	68%	52%	42%	76 (0.52%)	305 (0.4%)	1289 (0.46%)	15 (5%)	139 (46.3%)	182 (60.7%)	24	428	569
GradArgmax	69%	51%	45%	75 (0.51%)	317 (0.42%)	1245 (0.45%)	14 (4.7%)	144 (48%)	185 (61.7%)	26	687	894
Meta-attack	89%	72%	61%	25 (0.16%)	214 (0.32%)	752 (0.27%)	12 (4%)	89 (29.7%)	152 (50.7%)	28	1057	1752
EpoAtk	92%	74 %	64%	21 (0.14%)	195 (0.26%)	704 (0.25%)	11 (3.7%)	83 (27.7%)	145 (48.3%)	92	12963	20192
Our Attack	**97%**	**93%**	**94%**	**19 (0.13%)**	**155 (0.2%)**	**495 (0.18%)**	**8 (2.7%)**	**38 (12.7%)**	**48 (16%)**	**23**	**326**	**368**

and EpoAtk [6]. We choose these attacks because 1) a similar attack vector, i.e., adding/removing graph edges; 2) a similar threat assumption, i.e., knowing the model's gradient information to craft adversarial examples; 3) the similar applications, i.e., graph/node classification or embedding. To make a fair comparison, the attack baselines applicable to spatial-temporal graphs are developed as follows: 1) treat a spatial-temporal graph as a 3D graph without differentiating the time and space; 2) apply the three "spatial-only" 2D graph attack algorithms to the unfolded 3D graph and generate adversarial examples from the whole graph space. We only constrain the maximum number of changed edges (i.e., 1%) but no constraint for a number of the changed time-slices. As Table VII shows, *Ants* always presents much better attack effectiveness and stealthiness than the other three augmented attacks for all STGNNs. In addition, it also achieves **the lowest time cost among all solutions**, which aligns with our complexity analysis in Sec IV. It is noteworthy that, with the complexity of STGNNs increasing, the gaps of all metrics between our attack and others are significantly increased (e.g., for success rate gap, 5% in ST-GCN vs. 19% in AS-GCN vs. 30% in MS-G3D comparing with the best baseline using EpoAtk). This is because our attack optimizes the search of graph structure change by recognizing that the spatial and temporal dimensions of the model need to be considered separately, and constructing an optimization framework to search across these dimensions differently, consequently, making the process more efficient and crafted adversarial examples much effective and more indistinguishable.

VI. CONCLUSION

Spatial-temporal graph neural networks (STGNNs) have recently demonstrated remarkable performance in a variety of spatial-temporal applications. However, so far, studying the robustness of STGNNs to structure-based graph attacks is still in its infancy, in particular, by crafting new spatial-temporal graph adversarial attacks. This work strives to address this gap. We propose a spatial-temporal graph-based attack framework, namely the adaptive quantization-based spatial-temporal attack (*Ants*), to search adversarial examples in complex spatial-temporal graph data. Results show that *Ants* achieves competitive attack effectiveness and stealthiness in both white-box and black-box settings across various STGNNs on representative spatial-temporal tasks–traffic flow forecasting and human action recognition.

VII. ACKNOWLEDGEMENT

We thank all anonymous reviewers for their constructive comments and suggestions on this work. This work is partially supported by the National Science Foundation (NSF) under Grants No. CNS-2349538 and CCF-2401544.

REFERENCES

[1] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[2] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 922–929.

[3] J. Liu, N. Akhtar, and A. Mian, "Adversarial attack on skeleton-based human action recognition," *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[4] H. Wang, F. He, Z. Peng, T. Shao, Y.-L. Yang, K. Zhou, and D. Hogg, "Understanding the robustness of skeleton-based action recognition under adversarial attack," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 14 656–14 665.

[5] D. Zügner and S. Günnemann, "Adversarial attacks on graph neural networks via meta learning," *arXiv preprint arXiv:1902.08412*, 2019.

[6] X. Lin, C. Zhou, H. Yang, J. Wu, H. Wang, Y. Cao, and B. Wang, "Exploratory adversarial attacks on graph neural networks," in *2020 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2020, pp. 1136–1141.

[7] C. Song, Y. Lin, S. Guo, and H. Wan, "Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 01, 2020, pp. 914–921.

[8] M. Li, S. Chen, X. Chen, Y. Zhang, Y. Wang, and Q. Tian, "Actional-structural graph convolutional networks for skeleton-based action recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3595–3603.

[9] Z. Liu, H. Zhang, Z. Chen, Z. Wang, and W. Ouyang, "Disentangling and unifying graph convolutions for skeleton-based action recognition," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 143–152.

[10] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[11] Z. Wei, J. Chen, X. Wei, L. Jiang, T.-S. Chua, F. Zhou, and Y.-G. Jiang, "Heuristic black-box adversarial attacks on video recognition models," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, 2020, pp. 12 338–12 345.

[12] F. Liu, H. Liu, and W. Jiang, "Practical adversarial attacks on spatiotemporal traffic forecasting models," in *Advances in Neural Information Processing Systems*.

[13] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Security and Privacy (SP), 2017 IEEE Symposium on*. IEEE, 2017, pp. 39–57.

[14] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*. ACM, 2017, pp. 506–519.

[15] A. Shahroudy, J. Liu, T.-T. Ng, and G. Wang, "Ntu rgb+ d: A large scale dataset for 3d human activity analysis," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1010–1019.

[16] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev *et al.*, "The kinetics human action video dataset," *arXiv preprint arXiv:1705.06950*, 2017.

[17] C. Chen, K. Petty, A. Skabardonis, P. Varaiya, and Z. Jia, "Freeway performance measurement system: mining loop detector data," *Transportation Research Record*, vol. 1748, no. 1, pp. 96–102, 2001.

[18] X. Wei, J. Zhu, S. Yuan, and H. Su, "Sparse adversarial perturbations for videos," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 8973–8980.

[19] M. Sun, J. Tang, H. Li, B. Li, C. Xiao, Y. Chen, and D. Song, "Data poisoning attack against unsupervised node embedding methods," *arXiv preprint arXiv:1810.12881*, 2018.

[20] H. Dai, H. Li, T. Tian, X. Huang, L. Wang, J. Zhu, and L. Song, "Adversarial attack on graph structured data," in *International Conference on Machine Learning*, 2018, pp. 1123–1132.