TactileGCN: A Graph Convolutional Network for Predicting Grasp Stability with Tactile Sensors

1st Alberto Garcia-Garcia*
3D Perception Lab
University of Alicante
Alicante, Spain
agarcia@dtic.ua.es

2nd Brayan S. Zapata-Impata*
Automatics, Robotics and Artificial Vision Lab
University of Alicante
Alicante, Spain
brayan.impata@ua.es

3rd Sergio Orts-Escolano
3D Perception Lab
University of Alicante
Alicante, Spain
sorts@ua.es

4th Pablo Gil
Automatics, Robotics and Artificial Vision Lab
University of Alicante
Alicante, Spain
pablo.gil@ua.es

***Abstract*—Tactile sensors provide useful contact data during the interaction with an object which can be used to accurately learn to determine the stability of a grasp. Most of the works in the literature represented tactile readings as plain feature vectors or matrix-like tactile images, using them to train machine learning models. In this work, we explore an alternative way of exploiting tactile information to predict grasp stability by leveraging graph-like representations of tactile data, which preserve the actual spatial arrangement of the sensor's taxels and their locality. In experimentation, we trained a Graph Neural Network to binary classify grasps as stable or slippery ones. To train such network and prove its predictive capabilities for the problem at hand, we captured a novel dataset of ~ 5000 three-fingered grasps across 41 objects for training and 1000 grasps with 10 unknown objects for testing. Our experiments prove that this novel approach can be effectively used to predict grasp stability.**

I. Introduction

When we humans grasp objects, we know whether the grip is stable or not before lifting the object. It is not necessary to raise our hands in order to check such state of the grasp. Using our tactile sense, along with our vision and other senses, we can accurately predict the stability of the grasp. This skill is desirable for any robotic manipulator since it favors the early detection of grasp failures so the robot can react in consequence: for example, a re-stocking robot working in a store would recognize when an object could slip from its hand and, therefore, avoid breaking it.

The problem of predicting the stability of a grasp is a task under research in the field of robotic grasping. In order to approach a solution to it, tactile sensors are being used as the main source of data since they provide valuable information (e.g. temperature, pressure) about the acting forces during the interaction of the robotic hand with the objects [9]. As for the stability prediction, two states are usually distinguished: stable, meaning that the object is firmly grasped; or slippery, meaning that the object could slide from the hand.

*These authors contributed equally.

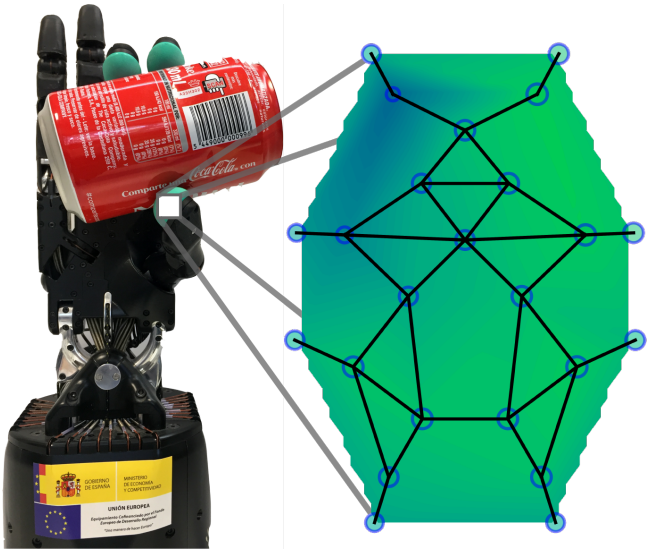

Fig. 1. In this work, we use a Shadow Dexterous hand equipped with three BioTac SP tactile sensors whose readings are transformed into graph representations. Those graphs are then fed as input to a GCN in order to learn to predict grasp stability.

Previous works found in the literature approach this problem following the next methodology: grasp the object, read the tactile sensors equipped in the fingers and/or palm of the hand, calculate custom features that try to characterize these two stability states and learn them in order to make future predictions [6], [13], [20], [22]. These proposals treat the tactile readings as classic signals: they pre-process them as if they were arrays, calculate features and learn their characteristics using probabilistic methods. As a consequence, their performance highly depends on the selected characteristics. Moreover, the spatial distribution inherent to the tactile sensor is lost due to the fact of squeezing the data into a one dimensional array.

In this work, we propose the use of Graph Neural Networks (GNNs) for predicting grasp stability. Since these are deep

learning models, there is no need to hand-engineer features because the algorithm is designed for learning them by itself. Moreover, graphs can reflect more accurately the real distribution of the electrodes in the sensor as well as their spatial relationships, which should be of great value for learning tactile features. The main contributions of this work can be summarized as follows:

- We process tactile readings using a novel perspective: instead of considering them as 1D arrays or 2D images, we build a 3D graph connecting the multiple sensing points (taxels) of the tactile sensor.
- We introduce a novel way of processing such information using Graph Neural Networks (GNNs).
- We quantitatively check the performance of this new methodology in the real world using a set of tactile sensors installed in a robotic hand, seen in Figure 1.
- We release an extension that effectively doubles the size of an already existing dataset [24] for grasp stability prediction and includes a whole new split for testing.

This paper is organized as follows. Section II reviews the state of the art of grasp stability prediction using tactile sensors and GNNs. Section III describes our system from the tactile graphs generation process to the network architecture. Section IV contains the methodology and data used to validate our proposal, as well as quantitative results to support our claims. Section V summarizes our findings and contributions. At last, Section VI states the main limitations of this work and draws some future research lines.

II. Related Works

In this section, we review the state of the art of the two main fields related to our work. On the one hand, we describe previous approaches for predicting grasp stability. On the other hand, we explain the most recent and relevant advances in neural networks for graph processing.

A. Grasp Stability Prediction

In the last years, deep learning models are being applied to the problem of grasp stability prediction using tactile sensors as input. Meier *et al.* [14] processed tactile readings using Fourier-related transforms and the resulting vectors were vertically stacked in order to create a matrix. Then, a Convolutional Neural Network (CNN) trained with these matrices learnt to predict stability. Although this approach used modern machine learning models, it still had to hand-engineer features.

In contrast, Cockbum *et al.* [5] proposed to use autoencoders to autonomously calculate the relevant characteristics for the task. Afterwards, a dictionary of basis features was built using a sparse encoding algorithm. Finally, the authors trained a Support Vector Machine (SVM) in order to predict grasp stability using the dictionary. Similarly, Kwiatkowski *et al.* [11] built a composite image by placing the readings of two matrix-like sensors side by side. Then, they used this tactile image as input for a CNN along with the proprioceptive data from the robot. As a result, the proposed method calculated by itself the features needed for predicting grasp stability.

A more recent trend suggests the interpretation of tactile sensors as images in order to exploit the potential of CNN as feature learners. In some cases, vision-based sensors are used for this purpose. Calandra *et al.* [3] used a tactile sensor that contained an internal camera, which recorded the deformation of the gel inside of the sensor throughout its contact with a surface. Then, the recorded tactile images were learnt using a CNN in order to predict the grasp outcome. In some other cases, the tactile sensor is not naturally arranged in an array or it does not contain a camera, so a pre-processing is necessary in order to get a tactile image. For example, Zapata-Impata *et al.* [24] studied how the readings from a non-matrix like sensor should be arranged in a matrix in order to train a CNN for grasp stability prediction. Although such approach showed promising results, the spatial distribution of the real sensor was not accurately reflected because it reduced the 3D locations of the taxels into 2D coordinates of a tactile image.

Recently, CNNs are being combined with Long Short-Term Memory Networks (LSTMs) for grasp stability prediction. Li *et al.* [12] in their work learnt visual features from a camera-based tactile sensor, similar to the one used by Calandra *et al.* [3], and an external camera pointing to the scene. These features were calculated using a pre-trained CNN. Then, both cameras features were concatenated and passed in time sequences to a LSTM, which was in charge of detecting slippage. Similarly, Zhang *et al.* [25] used another camera-based tactile sensor for grasp stability detection but in this work the authors trained a Convolutional LSTM (ConvLSTM) and they only passed the sensor images to the network.

B. Graph Neural Networks

Lately, GNNs have emerged as a solid alternative to process irregular data which can be structured as graphs. Their original focus was tasks whose data can be expressed as graphs holding locality, stationarity, and composionality principles in general. In the literature, various works have successfully made use of this kind of architecture to deal with unstructured 3D representations mainly in classification tasks. Most of them have proposed extensions to the well-known CNN architecture to process graph-structured data. That generalization is not trivial since various problems must be addressed when applying convolution filters in domains in which there is no regular structure. In that regard, there are two dominant ways to convolve a graph signal with a learned filter: spatial or spectral.

Spectral methods are characterized by providing a spectral graph theoretical formulation of CNNs on graphs using Graph Signal Processing (GSP) theory [18]. The fundamentals of this kind of methods rely on decomposing the graph Laplacian to form a Fourier basis via an eigendecomposition of the graph matrix, i.e., a spectral decomposition. By doing that, a convolution in the graph domain can be expressed as a multiplication in the spectral one. This kind of methods usually faces three challenges: the design of compactly supported filters, the definition of parameter sharing schemes among different graphs, and the aggregation of multi-scale infor-

mation. Arguably, the most common and limiting drawback is the first challenge: filters are not directly transferable to different graphs. Since filters are learned in the context of the spectrum of the graph Laplacian, a global graph structure must be assumed. In other words, only the signals on the vertices may change, the structure of the graph must remain the same.

Spatial methods constitute the straightforward generalization of convolutions to graph, just by sliding a filter on the vertices as a traditional CNN does with any other structured data representation. Despite its simplicity, the direct application of the definition of a convolution to graphs poses two difficulties: the definition of neighborhoods, and the ordering of the nodes to form receptive fields. Because of that, one common problem of spatial methods is the difficulty to generate a weight sharing schema across graph locations due to the fact that local neighborhoods can be completely different, i.e., the number of nodes adjacent to another one varies and there is no well-defined ordering for them.

Here we briefly review the most relevant GNNs that have been successfully applied to similar problems to the one at hand.

The pioneer spectral formulation of a CNN to operate over irregular domains modeled as graphs was introduced by Bruna et al. [2]. In that work, they exploited the global structure of the graph with the spectrum of its graph-Laplacian to extend the convolution operator. This method was applied to hand-written digit classification using the Modified National Institute of Standards and Technology (MNIST) dataset.

Defferrard *et al.* [7] proposed strictly localized filters, which are provable to be localized in a ball of a certain radius, i.e., hops from a specific vertex. That enhancement has some other collateral effects such as improved computational complexity for the filters (linear w.r.t. the support's size and the number of edges). They also introduced an efficient pooling strategy based on a rearrangement of the vertices as a binary tree. Their approach, namely *Chebyshev Spectral Graph Convolutional Operator* or just *ChebConv*, was successfully applied and performed similarly to classical CNNs in digits classification problems such as MNIST.

Kipf and Welling [10] introduced a set of simplifications to Bruna's [2] and Defferrard's [7] formulations to improve performance and scalability in large-scale networks. They proved the efficacy of their work on transductive node classification on very large scale networks for various problems such as semi-supervised document classification in citation networks (CiteSeer, Cora and PubMed datasets) and semi-supervised entity classification in a knowledge graph (NELL dataset). As its main feature, their *GCNConv* operator takes advantage of fast localized first-order features to achieve linear scaling in the number of graph edges.

Simonovsky and Komodakis [19], inspired by the idea from Jia *et al.* [3] about dynamic filter networks, took a similar approach for solving the weight sharing problem suffered by spatial methods. They introduced Edge-Conditioned Convolutions (ECCs) in which filter weights are conditioned on edge features and generated by a generator network. That generator, usually implemented as a Multi Layer Perceptron (MLP), outputs specific weights for each edge in the neighborhood. That method was successfully tested on point cloud classification problems (Sydney urban objects and ModelNet), a standard graph classification benchmarks, and also on MNIST.

Velickovic *et al.* [23] introduced a *Graph Attention* operator, namely *GATConv*, that leverages masked self-attentional layers to compute the hidden representations of each node in the graph, by attending over its neighbors, following a self-attention strategy. This approach addressed many of the key challenges of spectral-based methods and achieved or surpassed state of the art methods in the aforementioned citation network datasets as well as protein interaction ones.

Fey *et al.* [8] proposed the *Spline-based Convolutional Operator*, a continuous and spatial kernel that leverages B-spline bases's properties to efficiently filter graph data of arbitrary dimensionality. They prove this method to be successful in digit image graph classification problems using MNIST and graph node classification using the Cora dataset.

Following this success in those similar domains, we intend to use a GNN to process tactile sensor readings and predict grasp stability. By doing so, we expect that such architecture is able to better capture the spatial locality and relationships of the tactile sensor readings expressed as graphs instead of other non-spatial (1D arrays) or discrete (images) representations.

III. Proposal

In this section, we describe our full approach for predicting grasp stability using tactile sensors. The whole pipeline comprises three main components:

1) A robotic setup which consists of a Shadow hand and BioTac Sp sensors, all operated by Robot Operating System (ROS).
2) A tactile graph generator which takes the sensor readings and generates a proper graph representation for the network.
3) A GNN architecture to process such graphs and predict graph stability.

A. Robotic Set Up

In this work, we use the BioTac SP tactile sensors developed by Syntouch [21]. The sensor provides three different sensory modalities: force, pressure, and temperature. In more detail, this biomimetic sensor counts with 24 electrodes, also named taxels, integrated in just a single phalanx. These electrodes record signals from four emitters in the internal core of the sensor and, therefore, they measure the impedance in the fluid located between the internal core and the external elastic skin of the sensor. The fluid is displaced when the sensor makes contact with a surface, affecting that impedance read by the electrodes. Thus, the sensor can approximate how much pressure is being experienced at each electrode. In addition, the sensor features a hydro-acoustic pressure sensor in order to estimate a general pressure value and it also counts with a thermistor, which is used to detect vibrations and heat flows. The sensor is presented in Figure 2.

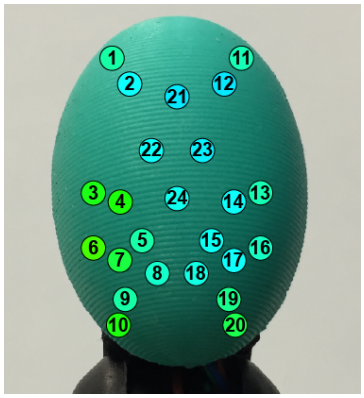

Fig. 2. BioTac SP tactile sensor with its 24 electrodes approximated position.

For our work, we use a setup of three BioTac SP sensors in the tip of the index, middle finger, and thumb of a Shadow Dexterous robotic hand developed by the Shadow Robot Company [17]. The Shadow hand is an anthropomorphic hand with five fingers and 20 Degrees of Freedom (DoF) in total. Those features allow the robot to reach a wide range of configurations that are comparable to those of a human hand. Its integration with the BioTac SP sensors is seamless since the sensor readings can be directly obtained using the ROS [15] framework, in which the Shadow hand works.

B. Tactile Graphs

In order to feed our Graph Neural Network, we expressed the aforementioned sensor readings in a novel graph representation, namely tactile graphs. Such graphs are triplet $G = (N, E, Y)$ where N is a set of 24 nodes $n_0, ..., n_{23}$ (one for each electrode or taxel in the sensor), E is a set of ordered pair of vertices called edges, and Y is the label or class of the graph (in our case, stable or unstable).

Each node n in the graph G represents a taxel and as such, they are characterized by a 3D position $p_n = (x_n, y_n, z_n)$ and a feature vector $f_n = (f_{n_0}, ..., f_{n_F})$ of arbitrary length F.

Node positions p_n are accurately mapped to the physical taxel (X, Y, Z) coordinates within the sensor. Such positions are specified in Table I. Edges or connections are generated following two different approaches: manual or k-Nearest Neighbors (k-NN). For the first approach, we manually specified undirected connections following proximity and symmetry criteria. For the second one, we generated directed edges towards each k-Nearest Neighbors for each node. Figure 3 shows a 3D graph representation of a tactile graph.

Node features f_n correspond to the taxel pressure readings. In the case of the most basic tactile graph, each node has three features, i.e., the pressure reading for each finger: index f_{n_0}, middle f_{n_1}, and thumb f_{n_2}. Figure 4 shows visualizations of the three components of the feature vector for sample graphs generated with various values of $k = 0$, $k = 2$, $k = 4$, $k = 8$.

C. Graph Neural Network

Our Graph Neural Network (GNN) of choice is based on the GCN model by Kipf and Welling [10]. Such model is

TABLE I
ACTUAL POSITION OF THE TAXELS INSIDE THE BIOTAC SP SENSOR EXPRESSED IN CARTESIAN COORDINATES (X, Y, Z) IN INCHES.

Taxel	X (inches)	Y (inches)	Z (inches)
1	0.386434851	-0.108966104	0.156871012
2	0.318945051	-0.205042252	0.120706090
3	0.087372680	-0.128562247	0.281981384
4	0.083895199	-0.235924865	0.201566857
5	-0.018624877	-0.300117050	0.094918748
6	-0.091886816	-0.120436080	0.284956139
7	-0.136659500	-0.237549685	0.187122746
8	-0.223451775	-0.270674659	0.071536904
9	-0.320752549	-0.199498368	0.127771244
10	-0.396931929	-0.100043884	0.151565706
11	0.386434851	-0.108966104	-0.156871012
12	0.318945051	-0.205042252	-0.120706090
13	0.087372680	-0.128562247	-0.281981384
14	0.083895199	-0.235924865	-0.201566857
15	-0.018624877	-0.300117050	-0.094918748
16	-0.091886816	-0.120436080	-0.284956139
17	-0.136659500	-0.237549685	-0.187122746
18	-0.223451775	-0.270674659	-0.071536904
19	-0.320752549	-0.199498368	-0.127771244
20	-0.396931929	-0.100043884	-0.151565706
21	0.258753050	-0.252337663	0.000000000
22	0.170153841	-0.274427927	0.072909607
23	0.170153841	-0.274427927	-0.072909607
24	0.075325086	-0.298071391	0.000000000

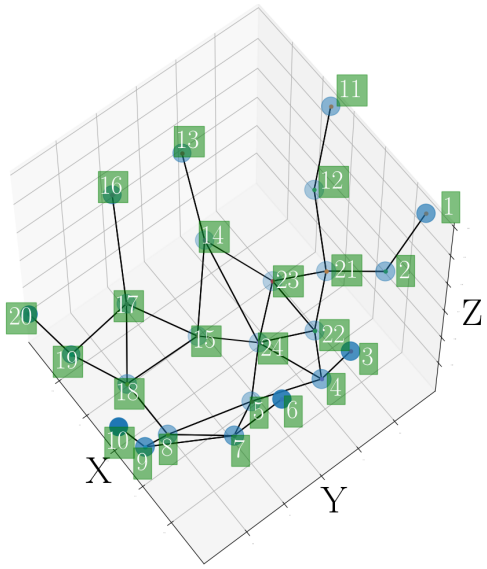

Fig. 3. 3D visualization of the tactile graph layout using the accurate spatial arrangement from the actual BioTac SP sensor. Graph edges correspond to the manually defined connections.

arguably one of the most successful, yet simple, approaches to date to generalize a well-established model such as the CNN to arbitrarily structured graphs [1] [16]. Their proposal, which is somewhat similar to Defferard's *et al.*, introduce a set of simplifications into a framework of spectral graph convolutions to make them train significantly faster and achieve state-of-the-art levels of accuracy across various classification tasks [7].

The goal of such models is to learn features on a graph $G = (N, E, Y)$ by taking as input a feature matrix X $(N \times F$

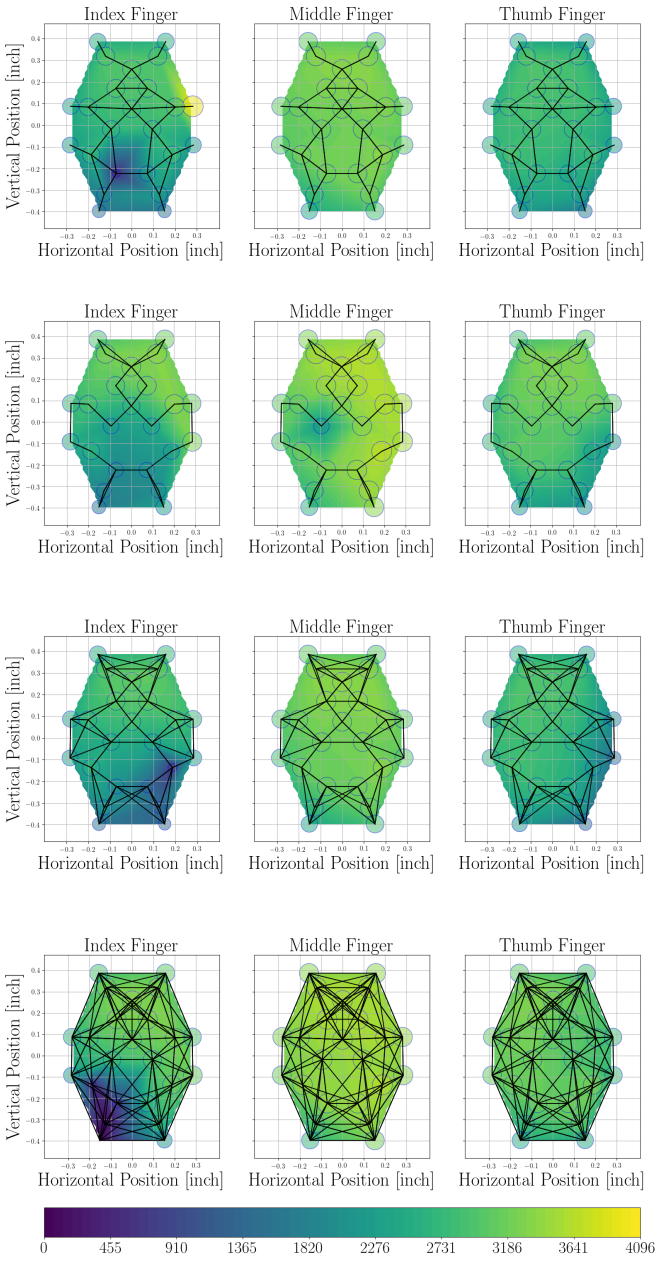

Fig. 4. From top to bottom, undirected tactile graphs generated with various k-NN configurations: $k = 0$ (manually defined edges), $k = 2$, $k = 4$, and $k = 8$. The three features (fingers) f_{n_0}, f_{n_1}, and f_{n_2} are deocupled into three different plots and represented as contour plots in the XY plane. Nodes or taxels are shown as blue semi-transparent circles whose size depends on the pressure read on them. Undirected edges are represented by black lines. Features are color-coded in the range $[0, 4096]$.

with a feature vector f_n for each node n) and a description of the graph structure in the shape of an adjacency matrix A (computed from the set of edges E in the graph). The output is another feature matrix Z ($N \times F'$ with node-level feature vectors f'_n with a predefined number of output features F').

Each GCN layer $H^(l)$ in a network with L layers can be expressed as a non-linear function $H^{(l+1)} = f(H^{(l)}, A)$. The first layer takes the input feature matrix ($H^{(0)} = X$) and the final layer generates the output node-level feature matrix ($Z = H^{(L)}$). Each intermediate layer generates a node-level feature matrix $Z^{(l)}$ which is fed to the next layer. In the case of Kipf and Welling [10], the graph-convolution layer $f(H^{(l),A)}$ is defined, in the most basic instantiation, as $\sigma(AH^{(l)}W^{(l)})$, where σ is an activation function of choice and $W^{(l)}$ is the weight matrix for the l layer.

This basic framework was heavily extended to overcome two limitations: (1) unless there are explicitly defined self-loops in the graph, the multiplication of A only sums up the feature vectors of all the neighboring nodes but not the node itself, and (2) since A is not normalized by default, the multiplication of A has a huge impact on the scale of the feature vectors. Overcoming those two limitations is crucial to improve the model's convergence.

In order to fix those two limitations, they first enforced self-loops in the graph by adding the identity matrix to A so the new adjacency matrix is $\hat{A} = A+I$. Secondly, they normalized that adjacency matrix in a row-like fashion by leveraging a symmetric normalization with the diagonal node degree matrix $\hat{D}$ of $\hat{A}$. Those two improvements combined form the layer propagation rule proposed by Kipf and Welling [10]: $f(H^{(l)}, A) = \sigma(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}H^{(l)}W^{(l)})$. This is the *GCNConv* operator that we used to build our GNN.

However, it is important to remark again that this model produces a feature matrix with node-level feature vectors yet our problem needs to classify the whole graph either as stable or slippery. To produce such binary graph-level classification output we need to introduce pooling operations to reduce the amount of nodes in the graph and/or fully connected layers to perform high-level reasoning.

IV. Experimentation

We conducted several experiments in order to validate our approach. In this section we describe the dataset we used to carry out such experiments. In addition, we provide all the details of our methodology to ensure the reproducibility of our procedures. At last, we discuss all the experiments that led us to the architecture described in the previous section.

A. Dataset

The dataset used in our experiments was first introduced in [24] as the *BioTac SP Images* dataset. It contains grasp samples performed over 41 objects with different geometries (i.e. cylinders, spheres, boxes), materials (i.e. wood, plastic, aluminum), stiffness levels (i.e. solid, soft) as well as sizes and weights. Those objects are shown in Figure 5. For this work, added 10 new objects with similar materials but different geometries and stiffness levels (see Figure 6). The original 41 were left for the training set whilst the new ones were separated into a test set. Both sets, training and test, were recorded following these steps:

1) **Grasp the test object:** the hand performed a three-fingered grasp that contacted the object with each of the fingers equipped with a tactile sensor.
2) **Read the sensors:** a single reading was recorded then from each of the sensors at the same time.
3) **Lift the object:** the hand was raised in order to lift the object and check the outcome.
4) **Label the trial:** the previously recorded tactile readings were labeled according to the outcome of the lifting with two classes (stable, i.e., it is completely static, or slip, i.e., either fell from the hand or it moves within it).

Fig. 5. The original training set of 41 objects.

Fig. 6. The newly captured test set of 10 objects.

There are two hand configurations in the original dataset: *palm down* grasps were performed pointing the palm of the hand downwards while *palm side* grasps were recorded pointing it to one side, with the thumb upwards. In this work, we have added a new configuration: *palm 45* which is in between the other two configurations at an angle of 45 degrees. Figure 7 shows the aforementioned hand configurations.

Table II provides a quantitative summary of the extended dataset for both splits and all configurations.

To the best of our knowledge, there is only one previous work that released a dataset of tactile recordings for the task of grasp stability detection, which is the BiGS dataset [4]. In their work, Chebotar *et al.* recorded 2000 grasps on three standing objects (a cylindrically-shaped box of wipes, a cubically-shaped box of candy and a ball) using a Barret three-fingered

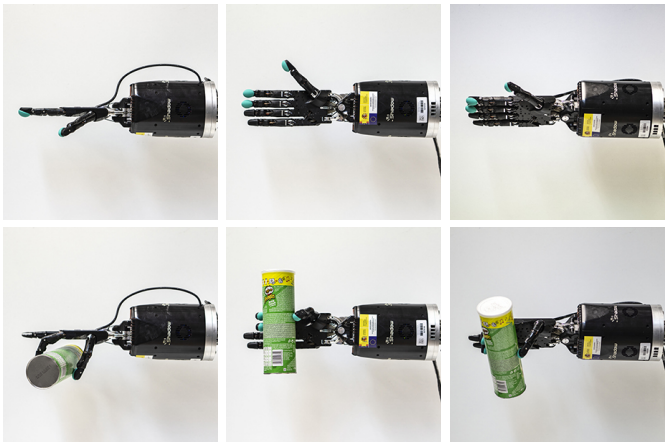

Fig. 7. (Top row) Samples of the three hand configurations in the dataset: (from left to right) *palm down*, *palm side*, and *palm 45*. (Bottom row) The same configurations but grasping an object.

TABLE II
SUMMARY OF THE EXTENDED BIOTAC SP DATASET WHICH WAS USED IN THIS WORK TO VALIDATE OUR GRAPH-BASED ARCHITECTURE.

	Training Set		Test Set	
Configuration	**Stable**	**Slippery**	**Stable**	**Slippery**
Palm Down	667	609	153	163
Palm Side	603	670	157	165
Palm 45	1058	1075	250	261
All	2328	2354	560	589

hand, which was equipped with three BioTac tactile sensors. Our work extends the BioTac SP dataset firstly introduced in [24], counting with more than 4000 training grasps and 1000 test grasps with three BioTac SP tactile sensors recorded using 51 objects and various orientations, both for the objects and the hand. The dataset is freely available at GitHub [1].

B. Experimental Setup

All experiments were run on a computer with an i7-8700 CPU 3.20 GHz (6 cores / 12 threads) with an Z370 chipset motherboard, 16 GiB DDR4 RAM 2400 MHz CL15, a Samsung SSD 860 EVO 250 GiB, and an NVIDIA Titan X Maxwell (12 GiB) GPU. Everything was implement in Python 3.6, PyTorch 0.4.1, PyTorch Geometric 0.3.1, CUDA 10.0 (with driver version 410.73).

For most experiments, we report accuracy as our main metric to iterate and draw conclusions over training and validation sets. For the test set, we report four different metrics: accuracy, precision, recall, and F1-score (the harmonic mean of precision and recall). To ensure generalization and give an accurate (and statistically correct) estimate of our prediction model performance we employ k-fold cross validation with $k = 5$. All reported results are the average of 10 rounds of 5-fold cross validation. For each cross-validation split, we train our models for 512 epochs using the ADAM optimizer.

[1] https://github.com/3dperceptionlab/biotacsp-stability-set-v2

The hyperparameters were chosen empirically as follows: 0.01 learning rate and $5e^{-4}$ weight decay.

The whole source code and dataset for this work can be downloaded from the corresponding GitHub repository[2].

C. Network Depth and Width

In these experiments, we investigate the impact of network depth (convolution layers) and width (amount of features per layer). To that end, we have tested ten different models ranging from one to ten *GCNConv* layers with increasing number of features (8, 16, 32, 48, 64). ReLU activations were used after each convolutional layer. Two fully connected layers were also placed at the end of the network (with 128 and 2 output features respectively) to produce the classification result. We made use of the manually defined graph connections ($k = 0$). Figure 8 shows the results of this set of experiments.

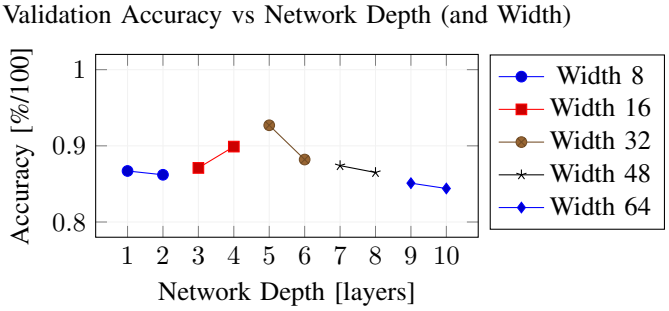

Fig. 8. Results of network depth and width study.

As we can observe, there is a dependency on both width and depth. Shallow networks tend to perform better than their deep counterparts. However, we can find a sweet spot on the architecture with 5 layers and 32 features ($8-8-16-16-32$). Shallower networks are not able to fully capture our problem while deeper ones tend to overfit our training data. Consequently, we will proceed with that network.

D. Graph Connectivity

For the connectivity experiments we took the previous best network and investigated the effect of graph connectivity. We experimented with manually specified edges ($k = 0$) and the k-NN strategy with $k = [1, 23]$. As shown in Figure 9, the performance of the network degraded as the connectivity of the graph increased in each experiment. Using the k-NN strategy, smaller k values achieved greater performance in terms of validation accuracy. However, none of them improved the performance (92.7%) yielded by the network trained with the graph created using the manual connectivity ($k = 0$).

In the manually created graph there are electrodes connected by an edge to just one other electrode, some others are connected up to four neighbors and the electrode in the center (24th electrode) is connected to six other points. As a result, there are different degrees of connectivity within the graph that could have given some insight to the network about the importance of each node in order to better learn the problem.

[2]https://github.com/3dperceptionlab/tactile-gcn

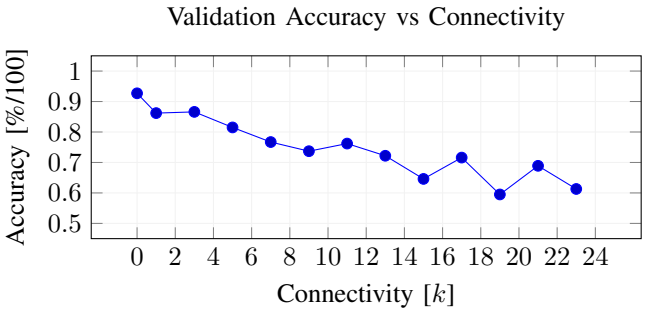

Fig. 9. Performance of the network according to the connectivity of the graph.

E. Generalization Tests

In order to prove the generalization capabilities of our system, we trained our best network ($8-8-16-16-32$ with $k = 0$) with our whole training set and evaluated it on the various test sets (palm down, palm side and palm 45). All results are reported in Table III.

TABLE III
RESULTS OF GENERALIZATION EXPERIMENTS ON THE TESTING SPLITS.

Test Set	Accuracy	Precision	Recall	F1
Down	0.741	0.741	0.751	0.745
45	0.774	0.774	0.783	0.778
Side	0.751	0.785	0.709	0.745

There is a significant drop in accuracy when dealing with completely unknown objects. Recall that the test set consists of new objects with different geometries and stiffness levels so they are substantially different from the training set. Taking all of this into account, and despite the difficulty of the testing set, we can expect gains from applying regularization and augmentation strategies to increase performance on data whose distribution is not that similar to the training set.

V. CONCLUSION

Tactile sensors provide useful information for robotic manipulation tasks like predicting grasp stability. Prior works in the literature tend to compute hand-engineered features that are later used for training a machine learning model. A recent trend process them as images, so deep learning techniques like CNNs can calculate relevant characteristics that lets the system distinguish a slippery grasp from a stable one. Inspired by this methodology, we propose in this work a novel approach to tactile data interpretation: we build a graph with the sensor's taxels because this structure keeps more accurately the spatial distribution and the local connectivity of these sensing points. The goodness of these properties and the tactile graph for grasp stability prediction were tested in experimentation.

We used three BioTac SP tactile sensors mounted in the tip of the index, middle and thumb of a Shadow Dexterous hand. In order to predict grasp stability using these graph representations of the tactile sensors, we trained a GCN with a custom dataset which was captured with more than 50 objects and 3 hand orientations. The robustness of the proposed system

was checked by testing the system with novel orientations and objects. In average, the GCN yielded a 92.7% validation accuracy on the prediction of grasp stability with novel objects or orientations.

VI. Limitations and Future Works

Given the obtained results, graph representations of tactile readings can be successfully used for learning the task of grasp stability prediction. Nevertheless, there are some drawbacks linked to their used. The first limitation of this proposal is the problem of defining the graph connectivity. We had to find a way of defining the location of the taxels as well as their connections in order to define the graph. In the case of using the tactile readings directly, none of this is necessary.

Moreover, GCN showed to be data hungry models for learning. In a previous work [24], the authors obtained higher validation rates (94.2%) with fewer data samples for training a CNN. For this work, it was necessary to capture more data in order to achieve similar accuracy rates in training. Furthermore, generalization to radically new objects has still a lot of room for improvement by leveraging techniques such as L2 regularization, dropout, or data augmentation itself.

As a future work, we also plan to decouple the currently unified GCN for the three fingers so that each graph is processed by a different network path. Furthermore, we plan to model the noise of each individual taxel and augment each sample on the fly by adding random noise following each taxel's distribution. At last, we plan to extend the architecture to predict grasp stability over temporal sequences by fusing the GCN model with LSTM networks.

Acknowledgment

Experiments were made possible by generous hardware donation from NVIDIA (Titan Maxwell). This work has been funded by the Spanish Government with Feder funds (TIN2016-76515-R and DPI2015-68087-R), by two grants for PhD studies (FPU15/04516 and BES-2016-07829), by regional projects (GV/2018/022 and GRE16-19) and by the European Commission (COMMANDIA SOE2/P1/F0638), action supported by Interreg-V Sudoe.

References

[1] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

[2] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

[3] Roberto Calandra, Andrew Owens, Manu Upadhyaya, Wenzhen Yuan, Justin Lin, Edward H. Adelson, and Sergey Levine. The Feeling of Success: Does Touch Sensing Help Predict Grasp Outcomes? In *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78, pages 314—-323, 2017.

[4] Yevgen Chebotar, Karol Hausman, Zhe Su, Artem Molchanov, Oliver Kroemer, Gaurav Sukhatme, and Stefan Schaal. BiGS: BioTac Grasp Stability Dataset. In *ICRA 2016 Workshop on Grasping and Manipulation Datasets*, 2016.

[5] Deen Cockbum, Jean-philippe Roberge, Thuy-hong-loan Le, Alexis Maslyczyk, and Vincent Duchaine. Grasp stability assessment through unsupervised feature learning of tactile images. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 2238–2244. IEEE, may 2017.

[6] Hao Dang and Peter K. Allen. Stable grasping under pose uncertainty using tactile feedback. *Autonomous Robots*, 36(4):309–330, apr 2014.

[7] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, pages 3844–3852, 2016.

[8] Matthias Fey, Jan Eric Lenssen, Frank Weichert, and Heinrich Müller. Splinecnn: Fast geometric deep learning with continuous b-spline kernels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 869–877, 2018.

[9] Zhanat Kappassov, Juan-Antonio Corrales, and Véronique Perdereau. Tactile sensing in dexterous robot hands — Review. *Robotics and Autonomous Systems*, 74:195–220, dec 2015.

[10] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[11] Jennifer Kwiatkowski, Deen Cockburn, and Vincent Duchaine. Grasp stability assessment through the fusion of proprioception and tactile signals using convolutional neural networks. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*, pages 286–292. IEEE, sep 2017.

[12] Jianhua Li, Siyuan Dong, and Edward Adelson. Slip Detection with Combined Tactile and Visual Information. 2018.

[13] Miao Li, Yasemin Bekiroglu, Danica Kragic, and Aude Billard. Learning of grasp adaptation through experience and tactile sensing. In *Intelligent Robots and Systems (IROS), 2014 IEEE/RSJ International Conference on*, pages 3339–3346. IEEE, sep 2014.

[14] Martin Meier, Florian Patzelt, Robert Haschke, and Helge J. Ritter. Tactile Convolutional Networks for Online Slip and Rotation Detection. In Alessandro E.P. Villa, Paolo Masulli, and Antonio Javier Pons Rivero, editors, *25th International Conference on Artificial Neural Networks*, volume 9887 of *Lecture Notes in Computer Science*, pages 12–19. Springer International Publishing, Cham, 2016.

[15] Morgan Quigley, Ken Conley, Brian P. Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. ROS: an open-source Robot Operating System. In *ICRA workshop on open source software*, volume 3, page 5, 2009.

[16] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, pages 593–607. Springer, 2018.

[17] Shadow Robot Company. Shadow Dexterous Hand, 2018. http://www.shadowrobot.com/products/dexterous-hand/.

[18] David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 30(3):83–98, 2013.

[19] Martin Simonovsky and Nikos Komodakis. Dynamic edgeconditioned filters in convolutional neural networks on graphs. In *Proc. CVPR*, 2017.

[20] Zhe Su, Karol Hausman, Yevgen Chebotar, Artem Molchanov, Gerald E. Loeb, Gaurav S Sukhatme, and Stefan Schaal. Force estimation and slip detection/classification for grip control using a biomimetic tactile sensor. In *Humanoid Robots (Humanoids), 2015 IEEE-RAS 15th International Conference on*, pages 297–303. IEEE, nov 2015.

[21] Syntouch. BioTac SP, 2018. https://www.syntouchinc.com/en/sensor-technology/.

[22] Filipe Veiga, Herke van Hoof, Jan Peters, and Tucker Hermans. Stabilizing novel objects by learning to predict tactile slip. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, volume 2015-Decem, pages 5065–5072. IEEE, sep 2015.

[23] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 1(2), 2017.

[24] Brayan S. Zapata-Impata, Pablo Gil, and Fernando Torres Medina. Non-matrix tactile sensors: How can be exploited their local connectivity for predicting grasp stability? *IEEE/RSJ IROS 2018 Workshop RoboTac: New Progress in Tactile Perception and Learning in Robotics*, abs/1809.05551, 2018.

[25] Yazhan Zhang, Zicheng Kan, Yu Alexander Tse, Yang Yang, and Michael Yu Wang. FingerVision Tactile Sensor Design and Slip Detection Using Convolutional LSTM Network. 2018.