Technical Disclosure Commons

Defensive Publications Series

September 2020

SEMI-SUPERVISED DEVICE TAG PREDICTION FOR AUTOMATIC NETWORK PROVISIONING

Xinjun Zhang

Dave Zacks

Qihong Shao

SEMI-SUPERVISED DEVICE TAG PREDICTION FOR AUTOMATIC NETWORK PROVISIONING

AUTHORS:
Xinjun Zhang
Dave Zacks
Qihong Shao

ABSTRACT

Device tagging is an important element in the world of network administration, offering an efficient way to organize network and computation resources (such as, for example, network devices, virtual machines, instances, etc.) and support efficient device provisioning and network segmentation (e.g., firewall rules, routing rules, etc.). The manual selection of tags and labelling of individual devices may be error prone and quite time consuming, particularly as the scale of a network grows. To address such challenges, techniques are presented herein that leverage aspects of Graph Convolutional Network (GCN) theory to offer a GCN-based approach for the accurate and automatic tagging of network devices employing a semi-supervised deep learning approach and requiring only minimal human expert knowledge (e.g., for training).

DETAILED DESCRIPTION

Device tagging is a network provisioning functionality for flexible network segments. Device tagging provides an important convenience within a range of network management tasks, including, for example, setting up virtual networks, policy administration and scalable grouping based on devices' roles, physical locations, security group, etc. It also enables efficient searching for a set of devices by tag name.

The manual selection and labelling of individual devices may be very time consuming and error prone, especially when as the scale of a network grows. Manual labelling can be very tedious and almost impossible when a network administrator or engineer tries to create a new tag for a large number of devices, or update existing tags for hundreds or thousands of devices. Even though scalable group tagging functionality is

available in some cases to facilitate tagging multiple devices in a batch, it can be very challenging for an inexperienced engineer to decide how to define rules to group devices together. A range of factors, including, for example, complicated network topology, heterogeneous network components, subnetworks located across many cities and countries, etc. make this task even more difficult.

To make the device tagging functionality more intelligent, automatic, and efficient, techniques are presented herein that employ a GCN-based approach to automatically tag network devices while only requiring a small amount of human expert knowledge as input (e.g., as few as one tagged device per group) to be able to predict tags for the rest of the devices in the network.

General tag prediction is a popular machine learning use case in many scenarios (including predicting merchandise tags for smart shopping, predicting movie and short video tags for personalized recommendation, and predicting customer tag for fashion recommendation), but one seldom sees the intelligent prediction of a device tag for network provisioning purposes.

As noted previously, device tagging is an efficient way to organize network and computation resources (e.g., network devices, virtual machines, instances, etc.) and enables efficient device provisioning and network segmentation (e.g., firewall rules, routing rules, etc.).

A typical device tagging use case would involve a network administrator or Subject Matter Engineer (SME) wanting to segment all network devices into N different groups. Traditionally, the engineer must manually label each device, one by one. Or, the engineer may first define some rule(s) and then he/she can group devices based on tags in which he/she defines a rule. By doing so, the network management portal/console automatically applies the tag to all devices that match the specified rule. Rules can be created based on criteria such as device name, device family, device series, IP address, location, or version. However, the definition of a rule requires a great deal of experience and knowledge about the network topology, all the device models, device families, and software/hardware versions. It is important to note that a rule may not assign the correct tags for all devices since the rule ‘draws’ a hard line between many different categories of devices and may inevitably make mistakes.

Currently, heuristic or intelligent tagging is seldom adopted in network provisioning or resource grouping process. In most cases for current products, a user must manually specify a tag value for each individual device. Such a limitation is obvious: a manual labelling process is very time consuming and requires a great deal of experience to assign a proper tag for each device. This can be a significant hurdle for engineers who lack sufficient experience to manage a huge network. In particular, when the network topology experiences a significant change and tagging rules needs to be subsequently changed, manual re-assigning tags to devices can result in a huge cost.

To address the challenges noted above techniques are presented herein that employ a novel deep learning approach (based on GCN theory), called GraphTag, which can automate network provisioning by predicting tags for each network device while only requiring a small amount of human input. Aspects of such techniques are designed explicitly to cope with the challenge of accurate and automatic tagging of a large number of network devices.

GCN is a neural network architecture for performing machine learning tasks on graph structured data (either directed or undirected graphs, see Figure 1, below) such as, for example, a wide area network (WAN) or a local area network (LAN), a social network, a scholar citation network, a drug-molecular interaction network, etc.

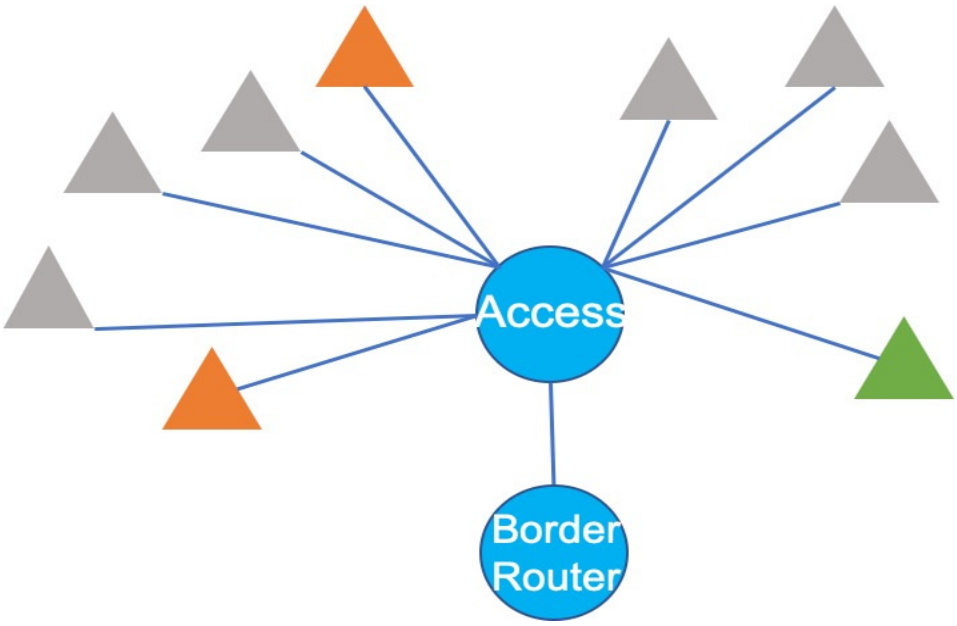

Figure 1: Simple Network Diagram

In such cases GCN has proven its capability and advantage in optimizing routing paths, predicting whether there is a potential connection between two persons, predicting whether the citation between an author and a publication is correct, predicting whether a drug can interact with a protein, etc. GCN also has performance that is comparable to a convolutional neural network (CNN) for computer vision tasks where the input data is an image.

GCN is similar to CNN due to the fact that filter parameters are typically shared over all locations in the graph (e.g., network). GCN learns the function of node features on a graph G=(V,E), where V denotes all of the nodes in the graph and E is the set of all of the edges in the graph. The inputs include two variables, X and A:

- X: a feature matrix describing every node x .
- A: the adjacent matrix of graph G which describes every edge in the graph.

GCN produces a node-level output, denoted as Z.

Similar to CNN, GCN performs convolution operations over each of the layers, and each layer's convolution operation is defined in an iterative way according to the formula:

$$H^{(l+1)} = f(H^{(l)}, A)$$

with $H^{(0)}$ =X and $H^{(L)}$ = Z, where L is the final output layer.

For illustration purposes the logical process of GraphTag using an example WAN network with 3 groups – Security, Office, and Internet of Things (IoT) – may be described with reference to Figure 2, below.

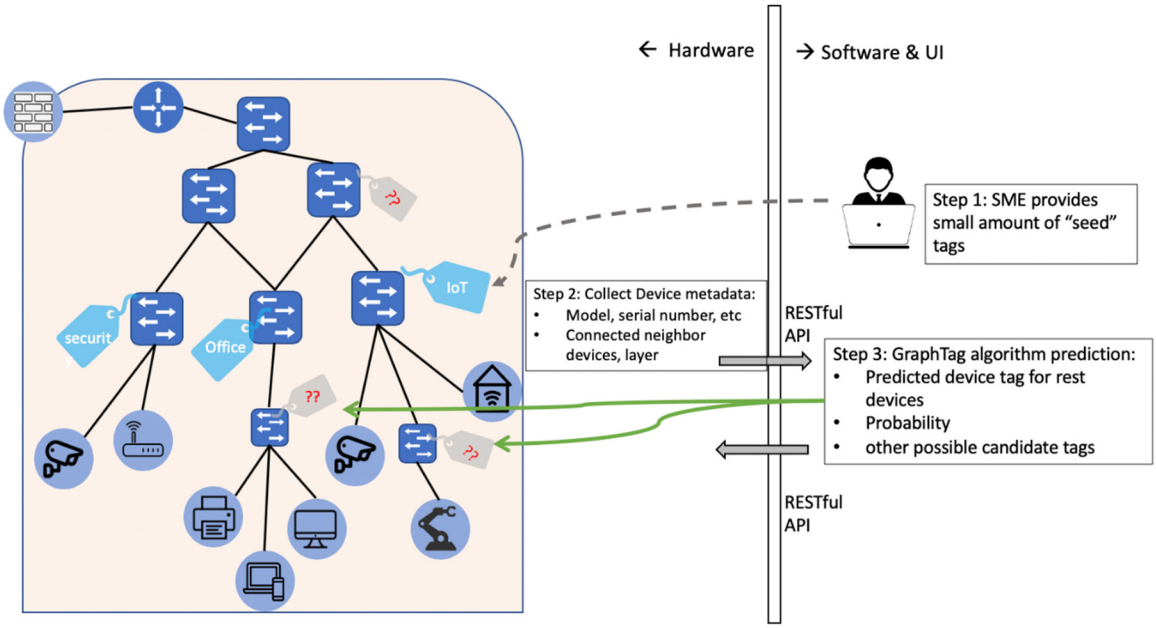

Figure 2: GraphTag – Logical Architecture View

As depicted in the above figure, a first component is a Seed Tag Provider (STP, see Step 1). A subject-matter expert (SME) or a network engineer manually assigns a tag (as a seed) for at least one device in each possible group. In this step, no rule is required to define a group. The SME just needs to pick one device that they are mostly confident about belonging to a specific group (e.g., Office, Security, or IoT).

A second component is a Network Metadata Collector (NMC, see Step 2 in Figure 2). Network features and the connections between each pair of devices are retrieved from a network management console. The features may include several dimensions such as, for example, network topology, device name, device family, device series, IP address, geographic location, software version, and hardware version. This ensemble of information may be passed to an algorithm within the techniques that are presented herein through an application programming interface (API).

A third component is a Network Tagging Predictor (NTP, Step 3 in Figure 2). In this component each dimension of information collected from a NMC may be imported into a model. The underlying algorithm is based on GCN. First, device and network features may be converted into a one-hot encoding vector, which then may be transformed through a GCN embedding layer into numerical vectors. The embedding vector for each

feature may be concatenated together into one single vector and fed into convolution layers for information aggregation and weighting. This algorithm is semi-supervised since it only requires a small amount of training data (e.g., tagged nodes in a network). After the algorithm is well trained, it can then predict labels/tags for all of the rest of the nodes. The predicted tags for rest nodes may be passed to a network console through an API.

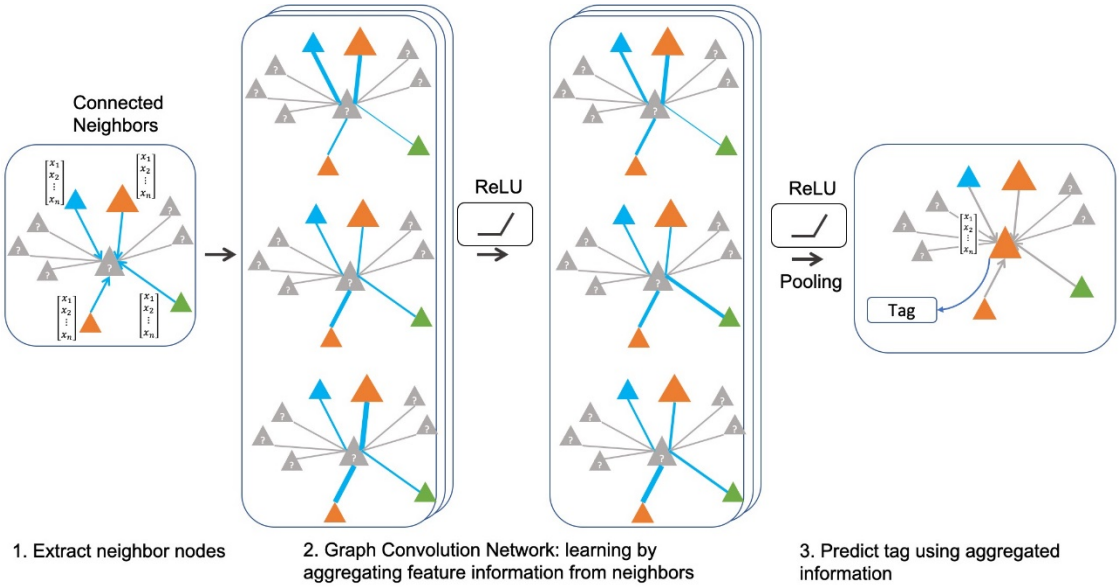

Figure 3: NTP Workflow and Architecture

The training of a NTP is implemented in three steps, as depicted in Figure 3, above. Those steps may include:

- In a first step, information about all of the neighbor nodes of the target node (in order to predict a tag/label) is collected and concatenated.
- In a second step, the concatenated information is embedded as a d-dimensional vector and processed through graph convolution operations to train the network. For an individual node, this step of information gathering and propagation through a convoluted layer is defined as:

$$h_{v_i}^{(l+1)} = \sigma\left(\sum_j \frac{1}{c_{ij}} h_{v_j}^{(l)} W^l\right) \quad (1)$$

where node x_i is encoded as vector v_i and $h^{(0)} = x_i$. j indexes the neighbor nodes of v_j. c_{ij} is a normalization factor for the edge (v_i, v_j). $W^{(l)}$ is a weight matrix for

the i-th network layer. And σ is a non-linear activation function. For all the input training data X, we can rewrite formula (1) into a matrix format:

$$H^{l+1} = f(H^l, A) = \sigma\left(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}H^l W^l\right) \quad (2)$$

where $H^{(0)} = X$, $H^{(l+1)}$ with dimension as R^{nxk}, X with dimension as R^{nxd}, and W with dimension as R^{dxk}. d is the dimension of features encoding a single node x_i, n is the total number of nodes in the network, and k is the total number of tag categories. A^ = A + I, where I is the identity matrix, and D^ is the diagonal node degree matrix of A^ and σ is the activation function, usually a rectified linear unit (ReLU) function.

- In a third step, after finishing the training process the weight matrix W is obtained through minimizing total training loss (usually cross entropy loss). To make a prediction of any node without a tag in the network, one may populate the node's feature vector value through the entire network and generate a predicted tag from the final layer.

The algorithm within the techniques that are presented herein returns the predicted tag together with a probability that a device should be assigned a certain tag. After a user executes the GraphTag algorithm and reviews a predicted device tag, the user is still able to change or edit the predicted results. For example, if the user believes that the predicted tag is wrong and/or the probability is low (e.g., less than 0.6) then they may change the tag based on their own judgement. As depicted in Figure 4, below, a user's editing or changing of a device tag may be recorded and sent back to the algorithm as feedback data to further improve the algorithm's accuracy.

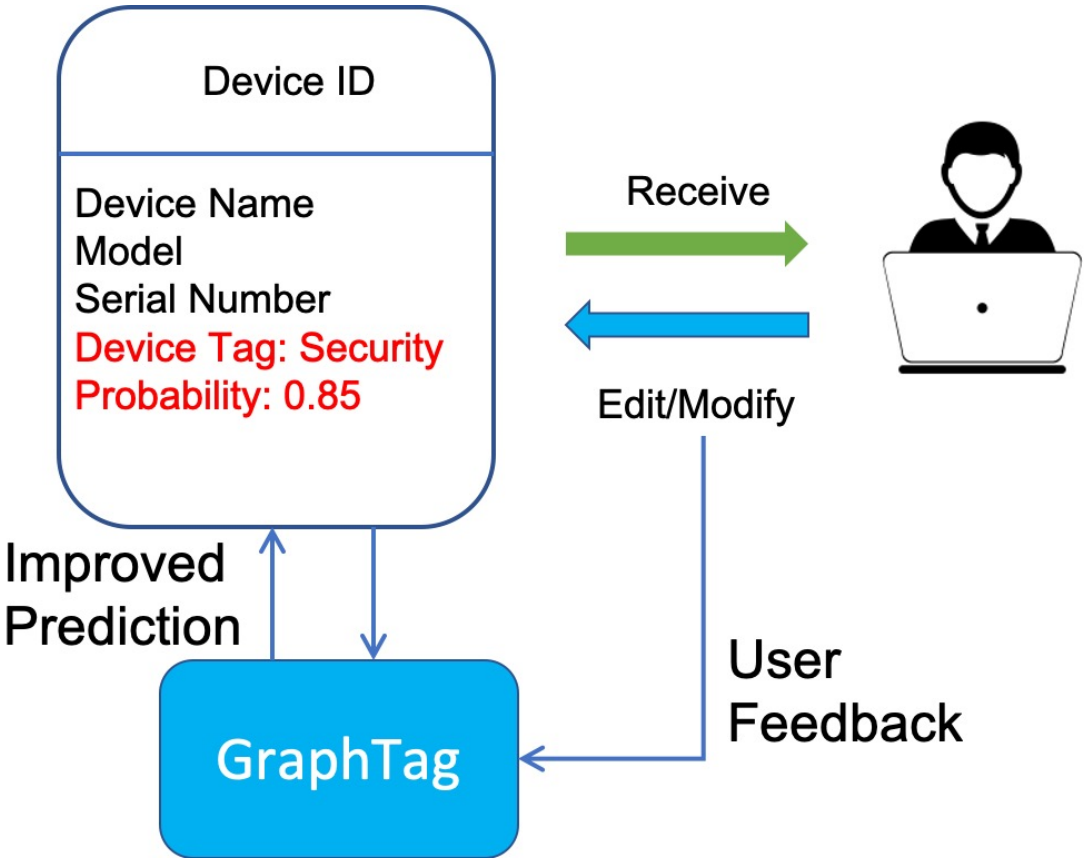

Figure 4: User Refining Predicted Device Tag

The techniques that have been presented herein offer a range of benefits including, for example:

- A GCN-based algorithm which is designed explicitly to cope with the challenge of accurate and automatic tagging of large amount of network devices.
- A semi-supervised deep learning approach (GCN) to automatically tag devices in a network.
- Efficiently tagging hundreds or thousands of devices while only requiring a small amount of training data (as few as one tagging per each group) from a human expert.
- The ability to work with any type of network topology and not be affected by upgrading/adding/removing devices in network.

In summary, the manual selection of tags and labelling of individual devices may be error prone and quite time consuming, particularly as the scale of a network grows. To

address such challenges techniques have been presented herein that leverage aspects of GCN theory to offer a GCN-based approach for the accurate and automatic tagging of network devices employing a semi-supervised deep learning approach and requiring only minimal human expert knowledge (e.g., for training).