GCNIIllustrator: Illustrating the Effect of Hyperparameters on Graph Convolutional Networks

Najdenkoska, I.; den Boef, J.; Schneider, T.; van der Werf, J.; de Ridder, R.; Fathurrahman, F.; Worring, M.

DOI
10.1145/3474085.3478566

Publication date
2021

Document Version
Final published version

Published in
MM '21

Citation for published version (APA):

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: https://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.

UvA-DARE is a service provided by the library of the University of Amsterdam (https://dare.uva.nl)

Download date:19 Apr 2024
ABSTRACT
An increasing number of real-world applications are using graph-structured datasets, imposing challenges to existing machine learning algorithms. Graph Convolutional Networks (GCNs) [2] are deep learning models, specifically designed to operate on graphs. One of the most tedious steps in training GCNs is the choice of the hyperparameters, especially since they exhibit unique properties compared to other neural models. Not only machine learning beginners, but also experienced practitioners often have difficulties to properly tune their models. We hypothesize that having a tool that visualizes the effect of hyperparameters choice on the performance can accelerate the model development and improve the understanding of these black-box models. Additionally, observing clusters of certain nodes helps to empirically understand how a given prediction was made due to the feature propagation step of GCNs. Therefore, this demo introduces GCNIllustrator - a web-based visual analytics tool for illustrating the effect of hyperparameters on the predictions in a citations graph.

1 INTRODUCTION
Many learning tasks benefit from the usage of datasets defined as graphs, since they encompass richer relational information compared to plain image or text. For instance, in recommendation engines graph-based systems can exploit the similarity between users and products profiles to make better recommendations. Social networks are a similar example which model the interaction between individuals or group of users. In publishing, citation networks are defined as graphs where papers are categorized based on similarity relationships between the scientific publications [6].

The analysis of such graphs is focused on tasks such as node classification, link prediction, clustering, etc. Graph Neural Networks (GNNs) are neural models designed to operate on graphs directly [6]. In the past years, many variants of the GNNs have been proposed, aiming to generalize well-established models, such as Convolutional Neural Networks (CNNs), to graph data. A state-of-the-art performance was achieved by Graph Convolutional Networks (GCNs) [2],
which generalized the convolutional operation from grid data, such as 2D images, to graph structures.

Naturally, GNNs and its variants come with a unique set of hyper-parameters compared to other neural models. For instance, GCNs have the ability to utilize graph edge weights during convolutional operations, which can highly impact the performance. Furthermore, GCNs have access to graph-specific preprocessing such as Graph Diffusion Convolution [3], which amplifies strong neighborhoods through means of signal diffusion in a given graph, potentially boosting accuracy. Having some of these specific hyperparameters and preprocessing modules is imperative to GCN performance. However, this comes at the cost of raising the entrance barrier due to the required technical knowledge in addition to the challenging hyperparameter tuning. Additionally, the feature propagation of GCNs through mean-pooling of the neighbourhood nodes, can be made more understandable through empirical analysis of visualized node clusters.

Having this in mind, this paper introduces GCNIllustrator, a web-based visual analytics tool which visualizes a citation graph and performs classification of scientific publications into seven categories by using user-controlled hyperparameters. GCNIllustrator is built to be a generic tool and it can be used with any graph-structured dataset. This tool targets machine learning engineers, data scientists, researchers, and anyone who wants to understand how hyperparameters affect GCN models in a visual and intuitive way, without worrying about CUDA compatibility and dependency issues.

2 GCN ILLUSTRATOR

The primary focus of GCNIllustrator is to let the user interactively explore different hyperparameters in GCNs, by retraining the model and observing the changes in the citation graph. Furthermore, the user can analyze how each paper was classified by hovering over the nodes, which displays the available metadata. Also, it can perform comparison to the baseline GCN [2], which yields a different view showing which nodes are correctly classified or misclassified by the configured model.

2.1 Dataset

The dataset used for this demo is the Cora citation dataset [4]. It contains 2708 nodes, denoting scientific publications, connected by 5429 edges denoting outgoing citations. Each sample belongs to one of seven categories of scientific publications, and are accompanied by 1433 one-hot encoded features for the absence or presence of certain words.

2.2 Training view

The first view of GCNIllustrator shows an interactive citation graph with the predictions of the baseline model, clustered into seven regions, as shown in Figure 1. The objective is to obtain more accurate predictions by using the input controls on the right menu and invoking the train feature which is essentially retraining the model in real-time with the new configuration. The graph can be further explored by using pinching gestures to zoom in or out and to navigate the graph. Also, hovering over a node displays the metadata in the left info card.

Visualizing a graph with a huge number of nodes is challenging and it often requires clutter reduction techniques. For this demo, we used force-directed approach [1, 5], which assigns forces among the set of edges and nodes, resulting in nodes from the same class being pulled together while nodes from different classes are pushed away. This approach allows generating an intuitive layout based solely on the provided list of nodes and edges. Also, it results in more symmetry and minimization of edge crossings, leading to a more aesthetic design. Each node has a different size, which denotes the confidence of the node classification, meaning that larger nodes are classified with a higher confidence.

2.3 Comparison view

The second view of GCNIllustrator, shown in Figure 2, is the comparison of the configured model and an optimized baseline GCN with hyperparameters copied from [2]. It displays the same graph, but with a different legend which distinguishes three types of nodes: grey when both the configured model and the baseline have the same predictions (both correct or wrong), green when the configured model is correct and the baseline is wrong and finally red when the configured model is wrong and the baseline is correct. This color scheme emphasises the nodes that are more challenging to be correctly classified and are usually missed by the configured models. By exploring the comparison view, the user can gain deeper insights in particular nodes such as outliers.

3 CONCLUSION

GCNIllustrator is a web-based visual analytics tool which aims to illustrate the effect of hyperparameters on GCNs performance, through an interactive visualization of a citation graph. It offers two major features, namely training a model with user-controlled hyperparameters and comparing it to a well-established baseline model.
REFERENCES


