Bachelor Research Project: Graph Machine Learning

Jannick Stuby
Department of Computer Science
© University of Stuttgart
Stuttgart, BW 70565
st160888@stud.uni-stuttgart.de

Christian Staib
Department of Computer Science
University of Stuttgart
Stuttgart, BW 70565
staib.christian@hotmail.de

Lukas Zeil
© Department of Computer Science
University of Stuttgart
Stuttgart, BW 70565
st161467@stud.uni-stuttgart.de

Abstract

We present our research on graph machine learning that we have made as part of the bachelor research project. We started our project examining DeepChem and MoleculeNet but later decided to focus on graph machine learning in general. During our project we had two main goals: to learn about graph machine learning in general and implement some graph machine learning algorithms ourselves. Therefore our report is structured in two parts: first we give an overview of graph machine learning and then we present our own implementations of some graph machine learning algorithms.

1 Introduction

1.1 Problem Domain

Neural Networks are thriving in the modern society with software ranging from facial recognition to language translation augmenting humans day-to-day. In this aspect, Machine Learning (ML) for chemical sciences gains more and more value and therefore is the focus of an increasing amount of research. In consequence, an increasing amount of data regarding molecules are gathered and labeled, multiplying the amount of information that can be used to train ML models to help chemical discovery and molecule analysis to aid chemical discovery for drug development and molecule investigation.

To aid in the improvement of ML models, benchmarks have been developed. A benchmark in a ML setting, is a site where models are compared against an accumulation of datasets, where the datasets are usually split into fixed training- and testing sets. The benchmark around which this project will be largely revolving is MoleculeNet. They tried a standardized approach for splitting their data, by creating more ways in which these datasets can be split, that can be applied to every dataset they have.

As the MoleculeNet paper was published in 2018, we try to find and use methods already used in other context but not in the paper. We will set up a dataset-to-metric model for each benchmark using new tools in four areas: splitters, featurizers, deep learning, and transfer learning. These results will be compared against the best-known results from MoleculeNet. While we may not be able to enhance the best-known results, we will formulate our accumulated knowledge for future exploration.

Preprint. Under review.

1.2 Definitions

A graph is a mathematical structure that consists of a set of vertices V and a set of edges E. Similarly a attributed graph is a mathematical structure that consists of a set of vertices and a set of edges, where each vertex and edge has an attribute, e.g. $A(V \bigcup E) \rightarrow \Sigma^*$. While this basic definition does not specify any conditions on the attributes, it is possible to limit the attributes to a certain set of values. Some graph machine learning methods require the attributes to be of certain types or to provide certain functionality.

1.3 MoleculeNet

TODO: add information on MoleculeNet and DeepChem framework

2 Related Work

As we started our project examining DeepChem, we first looked into handcrafted graph to vector methods.

Later we looked into graph machine learning.

2.1 Open Graph Benchmark

TODO: Add information on OGB and leaderboard

2.2 Graph Embedding

TODO: Add information on graph embedding relevant to problem domain

2.2.1 doc2vec

TODO: explain doc2vec approach with respect to problem domain, possibly move to another section (paragraph/word embedding)

2.2.2 graph2vec

TODO: explain graph2vec approach with respect to problem domain

2.3 Graph Transformers

TODO: add information for Graph Transformers and their usage for the problem domain and OGB leaderboards

2.3.1 Graphormer

TODO: explain Graphormer approach for OGB leaderboards

2.3.2 GraphGPS

TODO: explain GPS framework, recipes and focus on our uses

2.3.3 Heterogeneous Interpolation on Graphs

TODO: briefly explain and focus on our possible usage

3 Our approach

This is just a stub text to prevent the section from being empty.

In order to further strengthen our understanding, we decided to implement some graph machine learning methods ourselves. We had two ideas in mind: try something really simple and see if it

works, and try something more complex and see if it works. For the really simple method, we used random walk based methods.

3.1 Random walks based graph machine learning

A walk on graph is a unambiguously sequence of vertices and edges, where each vertex is connected to the next vertex by an edge. A random walk is a walk where the next vertex is chosen randomly from the set of vertices that are connected to the current vertex.

References

References follow the acknowledgments. Use unnumbered first-level heading for the references. Any choice of citation style is acceptable as long as you are consistent. It is permissible to reduce the font size to `small` (9 point) when listing the references. **Note that the Reference section does not count towards the eight pages of content that are allowed.**

[1] Alexander, J.A. & Mozer, M.C. (1995) Template-based algorithms for connectionist rule extraction. In G. Tesauro, D.S. Touretzky and T.K. Leen (eds.), *Advances in Neural Information Processing Systems 7*, pp. 609–616. Cambridge, MA: MIT Press.

[2] Bower, J.M. & Beeman, D. (1995) *The Book of GENESIS: Exploring Realistic Neural Models with the GEneral NEural SImulation System.* New York: TELOS/Springer–Verlag.

[3] Hasselmo, M.E., Schnell, E. & Barkai, E. (1995) Dynamics of learning and recall at excitatory recurrent synapses and cholinergic modulation in rat hippocampal region CA3. *Journal of Neuroscience* **15**(7):5249-5262.