Accelerating GNN Inference with GPU Feature Caching

by

Henry Liu

`henry.liu@utexas.edu`

Advised by Dr. Aditya Akella

A document submitted in partial fulfillment of the requirements for the degree of

Technical Report

at

University of Texas at Austin

Abstract

Graph Neural Networks (GNNs) have gained significant popularity due to their execellent performance in many graph representation learning tasks. This work investigates how GNN inference systems can respond to inference requests quickly without compromising prediction accuracy.

Existing GNN literature has identified that efficiently moving graph features to GPUs, known as data loading, is an important challenge at training time. To reduce data loading overheads, state of the art GNN training systems statically cache graph features on the GPU. We observe that the data loading problem is exacerbated at inference time and becomes a disproportionate bottleneck when serving inference requests. We improve upon existing static caching techniques to be better suited for inference.

In particular, we motivate the use of dynamic cache policies to exploit graph locality of inference requests and explore mechanisms to reduce the impact of dynamic cache overheads on latency and throughput. Our system targets a single-machine, multi-GPU setting and performs cache updates asynchronously to remove cache update operations from the latency-sensitive request-response path. We then leverage interconnects such as NVLink to allow GPUs to share a single logical cache and propose a lock-free synchronization mechanism to reduce potential lock contention due to cache updates.

Contents

Part I

A good part

You can also use parts in order to partition your great work into larger 'chunks'. This involves some manual adjustments in terms of the layout, though.

I Introduction

Graphs are flexible and powerful representations for data across many domains. Recently, Graph Neural Networks (GNNs) have emerged as a class of state-of-the-art (SoTA) machine learning methods for graph representation learning tasks. GNNs adopt techniques from traditional Deep Neural Networks (DNNs) and combine them with approaches that capture structural information about graphs.

Traditional DNNs have excelled in various tasks across domains such as computer vision [12][16] and natural language processing [6][13]. In these domains, inputs exhibit a fairly regular structure.

GNNs are deployed in production systems for many applications, including bio-informatics [25] [15], traffic forecasting [17] [3] [9], recommendation systems [24][23][8][18][22][4], cybersecurity [10] [14], and optimization [5][2], among many others. Although there has been significant work on GNN training systems [todo cite stuff], GNN inference is relatively understudied.

As opposed to traditional graph processing, graphs used with GNNs associate *features* with each node in the graph. Features are commonly represented as multidimensional tensors, often ranging from the hundreds to thousands. GNNs use these features to compute *embeddings* for each node in the graph by recursively aggregating each node's neighboring features. The resulting embeddings can be used for tasks such as node classification, link prediction, or graph classification. Different GNN architectures such as Graph Convolutional Networks (GCN) [11], GraphSAGE [7], and Graph Attention Networks (GAT) [21] perform different types of aggregations and operations on features, but share the same neighborhood aggregation principle.

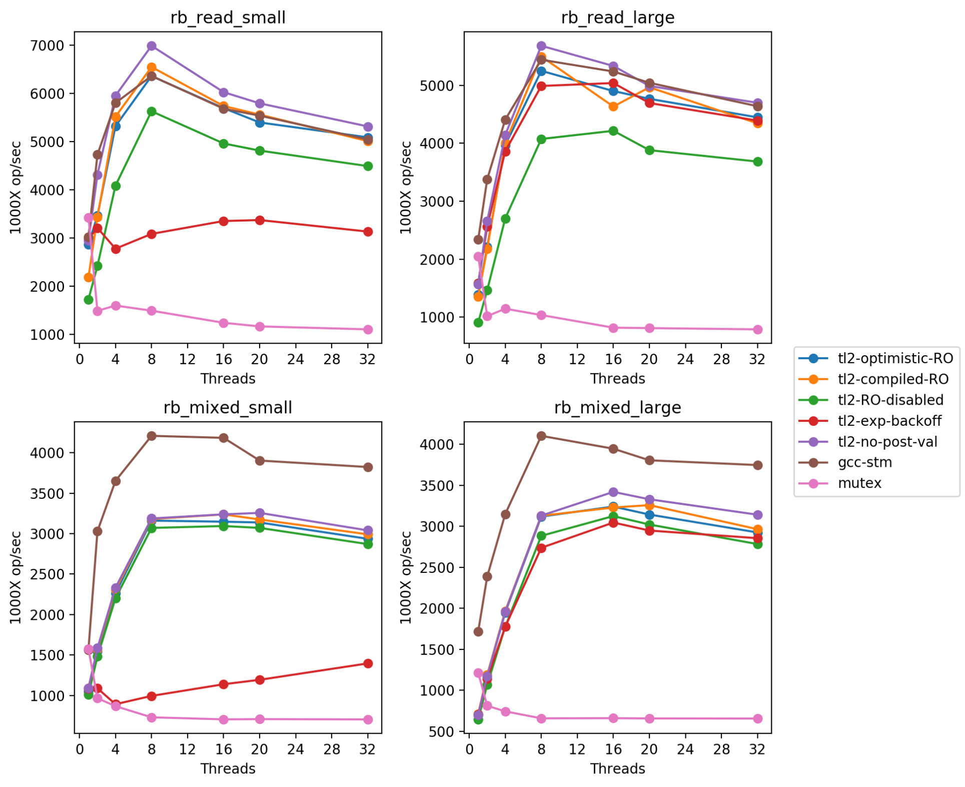

Figure 1.1: TL2 speedup on selected STAMP benchmarks

1.1 Contributions

In this work, we first identify the outsized impact of feature movement from CPU to GPU on GNN inference. Even when using static caching techniques borrowed from GNN training, this data loading step still remains a major bottleneck in achieving low latency inference.

1.2 Why?

I was not satisfied with the available templates for LaTeX and wanted to heed the style advice given by people such as Robert Bringhurst [1] or Edward R. Tufte [19, 20]. While there *are* some packages out there that attempt to emulate these styles, I found them to be either too bloated, too playful, or too constraining. This template attempts to produce a beautiful look without having to resort to any sort of hacks. I hope you like it.

1.3 How?

The package tries to be easy to use. If you are satisfied with the default settings, just add

```
\documentclass{mimosis}
```

at the beginning of your document. This is sufficient to use the class. It is possible to build your document using either LaTeX|, XeLaTeX, or LuaLaTeX. I personally prefer one of the latter two because they make it easier to select proper fonts.

1.4 Making this template *yours*

Prior to using this template, the first thing you want to do is probably a little bit of customisation. You can achieve quick changes in look and feel by picking your own fonts. With the `fontspec` package loaded and XeLaTeXor LuaLaTeXas your compiler, this is pretty simple:

```
\setmainfont{Your main font}
\setsansfont{Your sans-serif font}
\setmonofont{Your monospaced font}
```

Make sure to select nice combinations of that are pleasing to *your* eyes—this is your document and it should reflect your own style. Make sure to specify font names as they are provided by your system. For instance, you might want to use the following combination:

```
\setmainfont{Libre Baskerville}
\setsansfont[Scale=MatchLowercase]{IBM Plex Sans}
\setmonofont[Scale=MatchLowercase]{IBM Plex Mono}
```

If these fonts exist on your system, your normal text will look a little bit different from the other font used in this example PDF, while your sans-serif font will pair nicely with your `monospaced font`. You can also remove the `Scale` directive, but I find that most fonts pair better if they are adjusted in size a little bit. Experiment with it until you finds a combination that you enjoy.

Package	Purpose
`amsmath`	Basic mathematical typography
`amsthm`	Basic mathematical environments for proofs etc.
`babel`	Language settings
`booktabs`	Typographically light rules for tables
`bookmarks`	Bookmarks in the resulting PDF
`csquotes`	Language-specific quotation marks
`dsfont`	Double-stroke font for mathematical concepts
`graphicx`	Graphics
`hyperref`	Hyperlinks
`multirow`	Permits table content to span multiple rows or columns
`paralist`	Paragraph ('in-line') lists and compact enumerations
`scrlayer-scrpage`	Page headings
`setspace`	Line spacing
`siunitx`	Proper typesetting of units
`subcaption`	Proper sub-captions for figures

Table 1.1: A list of the most relevant packages required (and automatically imported) by this template.

1.5 Features

The template automatically imports numerous convenience packages that aid in your typesetting process. Table 1.1 lists the most important ones. Let's briefly discuss some examples below. Please refer to the source code for more demonstrations.

1.5.1 Typesetting mathematics

This template uses `amsmath` and `amssymb`, which are the de-facto standard for typesetting mathematics. Use numbered equations using the `equation` environment. If you want to show multiple equations and align them, use the `align` environment:

$$V := \{1, 2, \dots\} \tag{1.1}$$

$$E := \left\{(u, v) \mid \operatorname{dist}\left(p_u, p_v\right) \leq \epsilon\right\} \tag{1.2}$$

Define new mathematical operators using `\DeclareMathOperator`. Some operators are already pre-defined by the template, such as the distance between two objects. Please see the template for some examples. Moreover, this template contains a correct differential operator. Use `\diff` to typeset the differential of integrals:

$$f(u) := \int_{v \in \mathbb{D}} \operatorname{dist}(u, v) \, \mathrm{d}v \tag{1.3}$$

You can see that, as a courtesy towards most mathematicians, this template gives you the possibility to refer to the real numbers $\mathbb{R}$ and the domain $\mathbb{D}$ of some function. Take a look at the source for more examples. By the way, the template comes with spacing fixes for the automated placement of brackets.

1.5.2 Typesetting text

Along with the standard environments, this template offers `paralist` for lists within paragraphs. Here's a quick example: The American constitution speaks, among others, of (i) life (ii) liberty (iii) the pursuit of happiness. These should be added in equal measure to your own conduct. To typeset units correctly, use the `siunitx` package. For example, you might want to restrict your daily intake of liberty to 750 mg.

Likewise, as a small pet peeve of mine, I offer specific operators for *ordinals*. Use `\th` to typeset things like July 4th correctly. Or, if you are referring to the 2nd edition of a book, please use `\nd`. Likewise, if you came in 3rd in a marathon, use `\rd`. This is my 1st rule.

If you want to write a text in German and use German hyphenation rules, set the language of your text to german using `\selectlanguage{ngerman}`, or add

```
\PassOptionsToPackage{spanish}{babel}
```

before the `\documentclass` command to load a specific language. The languages `ngerman`, `french`, and `english` are loaded by default, with `english` being selected.

Quotation marks can be typeset using the `\enquote{...}` command from the `csquotes` package, which is preloaded by `latex-mimosis`. Depending on the currently selected language, quotes will look like “this”, „this“, or « this ». One must never use ”ASCII” quotation marks or even ’apostrophe’ symbols.

1.6 Changing things

Since this class heavily relies on the `scrbook` class, you can use *their* styling commands in order to change the look of things. For example, if you want to change the text in sections to **bold** you can just use

```
\setkomafont{sectioning}{\normalfont\bfseries}
```

at the end of the document preamble—you don't have to modify the class file for this. Please consult the source code for more information.

2 Background and Motivations

2.1 GNN Background

2.2 GNN Training Challenges

Neihgborhood explosion Feature transfer size GPU Sampling

2.3 GNN Inference Bottlenecks

2.4 Existing Caching Solutions

3 Implementation

3.1 System Overview

3.2 GNN Training Challenges

Neihgborhood explosion Feature transfer size GPU Sampling

3.3 GNN Inference Bottlenecks

3.4 Existing Caching Solutions

Acronyms

PCA	Principal component analysis
SNF	Smith normal form
TDA	Topological data analysis

Glossary

$\LaTeX$	A document preparation system
$\mathbb{R}$	The set of real numbers

Bibliography

1. R. Bringhurst. *The Elements of Typographic Style*. 4th ed. Hartley & Marks Publishers, Vancouver, British Columbia, Canada, 2012.

2. Q. Cappart, D. Chételat, E. B. Khalil, A. Lodi, C. Morris, and P. Veličković. "Combinatorial Optimization and Reasoning with Graph Neural Networks". In: *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*. Ed. by Z.-H. Zhou. International Joint Conferences on Artificial Intelligence Organization, 2021, pp. 4348–4355.

3. A. Derrow-Pinion, J. She, D. Wong, O. Lange, T. Hester, L. Perez, M. Nunkesser, S. Lee, X. Guo, B. Wiltshire, P. W. Battaglia, V. Gupta, A. Li, Z. Xu, A. Sanchez-Gonzalez, Y. Li, and P. Velickovic. "ETA Prediction with Graph Neural Networks in Google Maps". *CoRR* abs/2108.11482, 2021. arXiv: `2108.11482`. URL: `https://arxiv.org/abs/2108.11482`.

4. C. Gao, Y. Zheng, N. Li, Y. Li, Y. Qin, J. Piao, Y. Quan, J. Chang, D. Jin, X. He, and Y. Li. "A Survey of Graph Neural Networks for Recommender Systems: Challenges, Methods, and Directions". *ACM Transactions on Recommender Systems (TORS)*, 2022.

5. M. Gasse, D. Chételat, N. Ferroni, L. Charlin, and A. Lodi. "Exact Combinatorial Optimization with Graph Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Ed. by H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett. 2019, pp. 15554–15566. URL: `https://proceedings.neurips.cc/paper/2019/hash/d14c2267d848abeb81fd590f371d39bd-Abstract.html`.

6. A. Graves. "Generating Sequences With Recurrent Neural Networks". *CoRR* abs/1308.0850, 2013. arXiv: `1308.0850`. URL: `http://arxiv.org/abs/1308.0850`.

7. W. L. Hamilton, R. Ying, and J. Leskovec. "Inductive Representation Learning on Large Graphs". *CoRR* abs/1706.02216, 2017. arXiv: 1706.02216. URL: http://arxiv.org/abs/1706.02216.

8. X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang. "LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation". In: *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '20. Association for Computing Machinery, Virtual Event, China, 2020, pp. 639–648. ISBN: 9781450380164. DOI: 10.1145/3397271.3401063. URL: https://doi.org/10.1145/3397271.3401063.

9. W. Jiang and J. Luo. "Graph Neural Network for Traffic Forecasting: A Survey". *CoRR* abs/2101.11174, 2021. arXiv: 2101.11174. URL: https://arxiv.org/abs/2101.11174.

10. H. Kim, B. S. Lee, W.-Y. Shin, and S. Lim. "Graph Anomaly Detection With Graph Neural Networks: Current Status and Challenges". *IEEE Access* 10, 2022, pp. 111820–111829. DOI: 10.1109/ACCESS.2022.3211306.

11. T. N. Kipf and M. Welling. "Semi-Supervised Classification with Graph Convolutional Networks". *CoRR* abs/1609.02907, 2016. arXiv: 1609.02907. URL: http://arxiv.org/abs/1609.02907.

12. A. Krizhevsky, I. Sutskever, and G. E. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems*. Ed. by F. Pereira, C. Burges, L. Bottou, and K. Weinberger. Vol. 25. Curran Associates, Inc., 2012. URL: https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

13. G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer. "Neural Architectures for Named Entity Recognition". *CoRR* abs/1603.01360, 2016. arXiv: 1603.01360. URL: http://arxiv.org/abs/1603.01360.

14. Y. Liu, Z. Sun, and W. Zhang. "Improving Fraud Detection via Hierarchical Attention-Based Graph Neural Network". *J. Inf. Secur. Appl.* 72:C, 2023. ISSN: 2214-2126. DOI: 10.1016/j.jisa.2022.103399. URL: https://doi.org/10.1016/j.jisa.2022.103399.

15. M. Réau, N. Renaud, L. C. Xue, and A. M. J. J. Bonvin. “DeepRank-GNN: a graph neural network framework to learn patterns in protein–protein interfaces”. *Bioinformatics* 39:1, 2022.

16. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. “You Only Look Once: Unified, Real-Time Object Detection”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 779–788. DOI: 10.1109/CVPR.2016.91.

17. A. Roy, K. K. Roy, A. A. Ali, M. A. Amin, and A. K. M. M. Rahman. “SST-GNN: Simplified Spatio-temporal Traffic forecasting model using Graph Neural Network”. *CoRR* abs/2104.00055, 2021. arXiv: 2104.00055. URL: https://arxiv.org/abs/2104.00055.

18. J. Sun, Y. Zhang, W. Guo, H. Guo, R. Tang, X. He, C. Ma, and M. Coates. “Neighbor Interaction Aware Graph Convolution Networks for Recommendation”. In: *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR ’20. Association for Computing Machinery, Virtual Event, China, 2020, pp. 1289–1298. ISBN: 9781450380164. DOI: 10.1145/3397271.3401123. URL: https://doi.org/10.1145/3397271.3401123.

19. E. R. Tufte. *Envisioning information*. Graphics Press, Cheshire, CT, USA, 1990.

20. E. R. Tufte. *The visual display of quantitative information*. 2^{nd} ed. Graphics Press, Cheshire, CT, USA, 2001.

21. P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. *Graph Attention Networks*. 2018. arXiv: 1710.10903 [stat.ML].

22. J. Wu, X. Wang, F. Feng, X. He, L. Chen, J. Lian, and X. Xie. “Self-Supervised Graph Learning for Recommendation”. In: *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR ’21. Association for Computing Machinery, Virtual Event, Canada, 2021, pp. 726–735. ISBN: 9781450380379. DOI: 10.1145/3404835.3462862. URL: https://doi.org/10.1145/3404835.3462862.

23. L. Wu, J. Li, P. Sun, R. Hong, Y. Ge, and M. Wang. “DiffNet++: A Neural Influence and Interest Diffusion Network for Social Recommendation”. *IEEE Transactions on Knowledge and Data Engineering* 34:10, 2022, pp. 4753–4766. DOI: 10.1109/TKDE.2020.3048414.

24. R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec. “Graph Convolutional Neural Networks for Web-Scale Recommender Systems”. *CoRR* abs/1806.01973, 2018. arXiv: `1806.01973`. URL: `http://arxiv.org/abs/1806.01973`.

25. X.-M. Zhang, L. Liang, L. Liu, and M.-J. Tang. “Graph Neural Networks and Their Current Applications in Bioinformatics”. *Frontiers in Genetics* 12, 2021.