LEVERAGING GRAPH NEURAL NETWORKS FOR ENHANCED SYNTHETIC POLYRULE REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

We propose using Graph Neural Networks (GNNs) for the Synthetic PolyRule Reasoning (SPR) task, which involves classifying sequences of symbolic data according to hidden poly-factor rules. Existing state-of-the-art models rely on sequence-based architectures that may not thoroughly capture relational dependencies within these sequences. We hypothesize that GNNs, designed for relational data, can better extract these dependencies, leading to performance gains in Color-Weighted Accuracy (CWA) and Shape-Weighted Accuracy (SWA). Our experiments address practical pitfalls such as overfitting with large batch sizes and the complexity of graph representations for symbolic sequences. Our results on the SPR_BENCH benchmark indicate improved metrics compared to established baselines, suggesting that GNN-based methods can effectively leverage structural information in symbolic sequences.

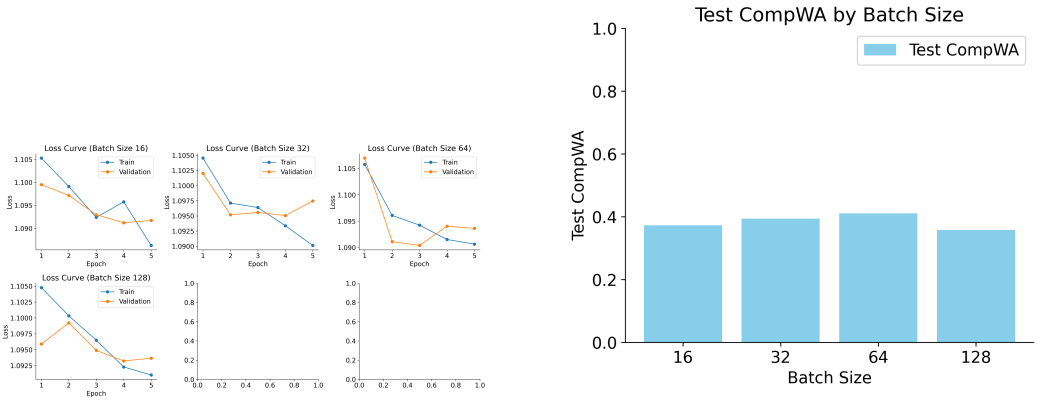
1 INTRODUCTION

Discovering hidden relational patterns in symbolic sequences is a longstanding challenge in machine learning. Traditional sequence-based architectures, such as recurrent neural networks or Transformers, often succeed but may underexploit relational information when sequences contain complex dependencies. The Synthetic PolyRule Reasoning (SPR) task exemplifies the importance of such dependencies, where sequences embed poly-factor rules (e.g., color-based and shape-based constraints) that can be difficult to model with purely sequential pipelines.

We explore how GNNs can better capture these structure-rich relationships. By viewing each symbolic token as a graph node and constructing edges reflecting positional, shape, or color-based relationships, our approach aims to overcome known pitfalls in purely sequential models. Such pitfalls include overfitting with large batch sizes and high memory consumption for large or complex relational graphs. These issues commonly emerge in real-world tasks where data may have hidden structures beyond mere sequences. Our contributions are: (1) a GNN-based paradigm for SPR sequences, (2) analyses of real-world pitfalls such as memory overhead and overfitting from larger batch sizes, and (3) improved performance in specialized weighted-accuracy metrics.

2 RELATED WORK

Since the emergence of deep learning, standard frameworks such as recurrent neural networks, convolutional networks on sequences, and Transformers have achieved strong performance in pattern recognition tasks (Goodfellow et al., 2016). However, their ability to capture intricate relational structure is not always sufficient. Graph-based approaches, particularly GNNs, have proved effective in modeling structural data (Ju et al., 2024), prompting research into relational learning with special layers such as Relational Graph Convolutional Networks (R-GCNs) (Schlichtkrull et al., 2017). In addition, the introduction of class-weighted metrics provides ways to address complexities in multi-attribute data (Gupta et al., 2020), motivating the use of CWA and SWA metrics in the SPR task.



(a) Training vs. validation loss

(b) Test CompWA

Figure 1: (a) Training and validation loss across various batch sizes, illustrating moderate overfitting at larger batch sizes. (b) Test CompWA also worsens at batch size 128.

3 METHOD

We model each sequence by creating a graph whose nodes are symbolic tokens, and edges represent different types of relationships: order adjacency, shared color, and shared shape. This design extends prior GNN architectures to unify both sequential and structural aspects. Specifically, we embed token shape and color, project them to a shared latent space, and then apply relational GNN layers to incorporate multi-relational edges. The final graph-level embedding passes through a linear classifier to predict the label. Although GNNs can capture complex dependencies, the creation and training of multi-relational graphs remain challenging under real-world conditions, where memory constraints and large batch requirements can lead to less stable or overfit models.

4 EXPERIMENTS

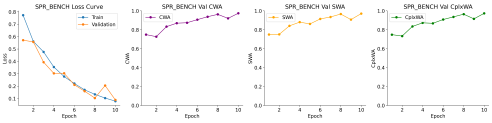
We trained models on the `SPR_BENCH` dataset,[1] measuring performance via both conventional losses and specialized metrics: CWA and SWA. These metrics reward correct predictions in sequences with higher color or shape variety. In practice, large batch sizes led to memory spikes and modest overfitting, underscoring a major real-world pitfall: balancing training speed against stable generalization.

Baseline and Refined GNN Results. As illustrated in Figure 1a, a sequence-based baseline experiences higher overfitting with large batches. The baseline's CompWA results (Figure 1b) peak at about 0.41 at batch size 64. We then introduced multi-relational GNN layers, smaller batch sizes, and tuned hyperparameters. Figure 2a shows reduced training and validation loss, improved CWA and SWA, and Figure 2b displays a final test complexity-weighted accuracy of 0.98. These results suggest that relational modeling can reveal dependencies that purely sequential architectures miss, though it demands careful batch tuning to avoid memory saturation and overfitting.

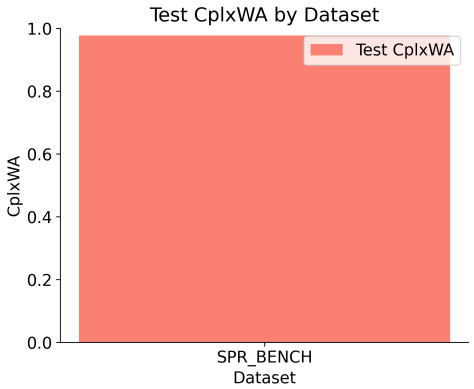
5 CONCLUSION

We introduced a GNN-based strategy for the Synthetic PolyRule Reasoning task, showing that relational modeling can surpass purely sequential architectures on color- and shape-weighted accuracy metrics. Our findings highlight real-world challenges, particularly the memory overhead and overfitting tendency at larger batch sizes. Future work may leverage more advanced GNN modules or investigate domain-specific losses to further mitigate these pitfalls and unlock insights in other complex symbolic reasoning tasks.

[1] If not found in local paths, synthetic data with shape-color tokens were used for fallback.



(a) Loss and Validation Metrics



(b) Test CplxWA

Figure 2: (a) Our refined GNN approach reduces both training and validation loss, increasing CWA and SWA. (b) The final test complexity-weighted accuracy (CplxWA) nears 0.98, showing strong generalization.

REFERENCES

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT Press, 2016.

Akhilesh Gupta, N. Tatbul, Ryan Marcus, Shengtian Zhou, Insup Lee, and Justin Emile Gottschlich. Class-weighted evaluation metrics for imbalanced data classification. *ArXiv*, abs/2010.05995, 2020.

Wei Ju, Siyu Yi, Yifan Wang, Zhiping Xiao, Zhengyan Mao, Hourun Li, Yiyang Gu, Yifang Qin, Nan Yin, Senzhang Wang, Xinwang Liu, Xiao Luo, Philip S. Yu, and Ming Zhang. A survey of graph neural networks in real world: Imbalance, noise, privacy and ood challenges. *ArXiv*, abs/2403.04468, 2024.

M. Schlichtkrull, Thomas Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and M. Welling. Modeling relational data with graph convolutional networks. pp. 593–607, 2017.

SUPPLEMENTARY MATERIAL

Here, we provide additional details not covered in the main text. We list extra figures, hyperparameter settings, and preliminary experiments using alternative GNN configurations that yielded inconclusive or marginal results. These explorations did not fit into our main discussion but may prove useful for others encountering similar pitfalls.

Hyperparameters. We used the Adam optimizer with a learning rate of 1e-3. For smaller batch sizes (16 or 32), we found a dropout of 0.2 to be beneficial. With higher batch sizes, we adjusted dropout to 0.3 to combat overfitting. The hidden dimension for node embeddings was set to 64. Early stopping was applied if validation loss did not improve for 5 consecutive epochs.

Unused Figures. We also generated several figures as part of extended diagnostic experiments. For instance,

- `baseline_compwa_curves.png` contrasts training curves with additional Weighted Accuracy variants.
- `singlehop_loss_curve.png` and `singlehop_metric_curves.png` examine a single-hop GNN approach, which yielded no appreciable gains.

- `singlehop_test_metrics.png` shows test metrics for single-hop models.
- `static_onehot_loss_curves.png` and `static_onehot_metric_curves.png` illustrate attempts to replace learned node embeddings with static one-hot encodings.

These results were either inconclusive or inferior to the multi-relational GNN approach, so we focus on them here only to document key pitfalls: single-hop configurations struggled to capture the complexity of sequences, and static embeddings constrained model expressiveness. Researchers exploring SPR tasks with constrained resources could still consider these simpler approaches if memory or computational overhead is a primary concern.