LEVERAGING GRAPH NEURAL NETWORKS FOR ENHANCED SYNTHETIC POLYRULE REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

We propose the use of Graph Neural Networks (GNNs) for the Synthetic PolyRule Reasoning (SPR) task, which involves classifying sequences of symbolic data according to hidden poly-factor rules. Current approaches rely on purely sequence-based architectures that may not capture the underlying relational and structural dependencies present in these sequences. We hypothesize that GNNs, with their ability to model relational data, can better capture these dependencies. Our experiments on the SPR_BENCH dataset indicate that a simple GCN-based model can attain competitive accuracy metrics and color-/shape-weighted accuracy scores, yet suffers from overfitting and class imbalance, highlighting the need for careful design choices.

1 INTRODUCTION

Devising robust models for reasoning over symbolic sequences is critical for practical applications like rule-based reasoning, semantic parsing, and combinatorial generalization (Goodfellow et al., 2016; **?**). In the Synthetic PolyRule Reasoning (SPR) task, sequences of tokens encode shapes and colors, and the classification label depends on a hidden set of poly-factor rules. Traditional sequence models, including LSTMs or Transformers, focus on token order but may overlook the inherent relational structure within such data. We explore Graph Neural Networks (GNNs) (**?**), hypothesizing that graph-based representations help capture dependencies such as token symmetry or shared attributes that extend beyond simple adjacency in a sequence.

Despite the promise of GNNs, our results indicate notable challenges. Although we observe performance improvements in certain metrics, our analyses show that the model often overfits to the training data, as indicated by rising validation loss and skewed confusion matrices. These insights underscore the complexity of harnessing GNNs for symbolic reasoning tasks and can guide future research toward mitigating overfitting and better handling class imbalance.

2 RELATED WORK

Symbolic reasoning tasks historically rely on model families that treat data as simple sequences, such as RNNs and LSTMs, or adopt attention-based mechanisms like Transformers (Goodfellow et al., 2016; **?**). However, symbol sequences often encode relational dependencies, prompting attempts to leverage graph-based structures to handle these connections. Early work on GNNs and their design principles can be found in (**?**), illustrating how graph-based learning can capture topological relationships. This paper expands such insights by using a GNN architecture specialized for SPR, a domain where relationships between shapes and colors are crucial factors in classification decisions.

3 BACKGROUND

The SPR task concerns classifying sequences composed of symbolic tokens representing shapes and colors. Each sequence is assigned a binary label based on hidden poly-factor rules. We examine two metric variants: Color-Weighted Accuracy (CWA) and Shape-Weighted Accuracy (SWA). They weight correctness by the complexity or diversity of shapes and colors in each sequence. We ob-

serve that relying purely on conventional accuracy can obscure poor performance on more complex samples.

4 METHOD

To capture relational structure, each sequence is represented as a graph. Tokens constitute nodes with concatenated one-hot encodings for shape and color. We connect consecutive tokens and optionally augment edges to encode additional relationships. A two-layer Graph Convolutional Network (GCN) with global average pooling is then trained on this graph representation to predict labels. The cross-entropy loss is minimized using Adam. We track both plain accuracy and complexity-weighted metrics derived from shapes and colors.

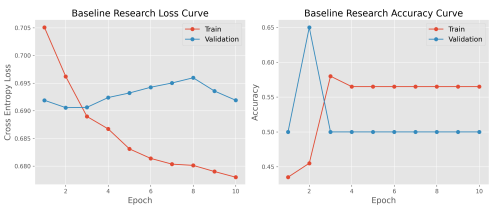
5 EXPERIMENTAL SETUP

We use the SPR_BENCH dataset,[1] partitioned into training, development, and test splits. All shapes and colors from the training partition form dictionaries reused for dev/test. For each split, we build graphs in `torch_geometric Data` objects. We train for up to 10 epochs, selecting hyperparameters by dev-set performance. To measure overfitting and class skew, we inspect external metrics beyond plain accuracy, such as confusion matrices.

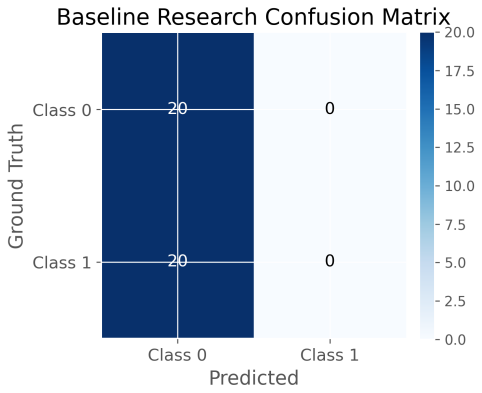
6 EXPERIMENTS

Quantitative results reveal a training accuracy of 58%, while validation accuracy reaches 65%. The corresponding losses are 0.678 (train) and 0.6906 (validation), showing a gap that suggests potential overfitting. We also observe that the model struggles with certain classes, predicting predominantly one label. As illustrated in Figure 1, the training and validation curves reveal a clear pattern of overfitting, while the confusion matrix indicates a strong bias toward predicting one class.

In addition, we explored multiple design variants (Depth-1 GCN, directed-edge GCN, fully connected GCN, sequence-order shuffled input, shape-only edges, etc.). The results from these variants, presented in the Appendix, reveal similar overfitting behaviors across architectures, suggesting that these pitfalls are not simply a consequence of the baseline model.



(a) Baseline Loss and Accuracy



(b) Baseline Confusion Matrix

Figure 1: (a) Training and validation curves for cross-entropy loss and accuracy highlight overfitting; (b) A confusion matrix dominated by one class prediction demonstrates imbalance.

[1]Detailed code for data loading can be found in the appendix.

7 CONCLUSION

We presented a GNN-based approach for the Synthetic PolyRule Reasoning task, offering a relational perspective that shows promise in capturing structural dependencies overlooked by purely sequential methods. Although we observe respectable validation accuracy and partial gains in color- and shape-weighted metrics, issues with overfitting and class imbalance remain limiting factors. Future investigations should explore advanced graph-based mechanisms, data augmentation, and more balanced training strategies to fully realize the potential of GNNs for symbolic reasoning tasks.

REFERENCES

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

SUPPLEMENTARY MATERIAL

A IMPLEMENTATION DETAILS

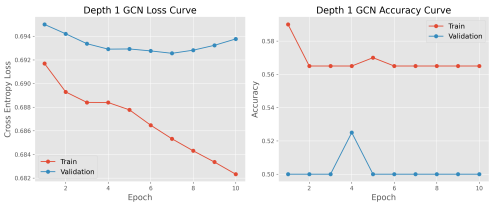
Below is an abbreviated snippet illustrating how data are processed. Full code includes hyperparameters, data augmentation steps, and training loops:

```
dataset = load_spr_bench(...)
graphs = []
for seq, label in dataset:
    node_features = encode_tokens(seq)
    edges = build_edges(seq, use_relational=True)
    gdata = Data(x=node_features, edge_index=edges, y=label)
    graphs.append(gdata)
```

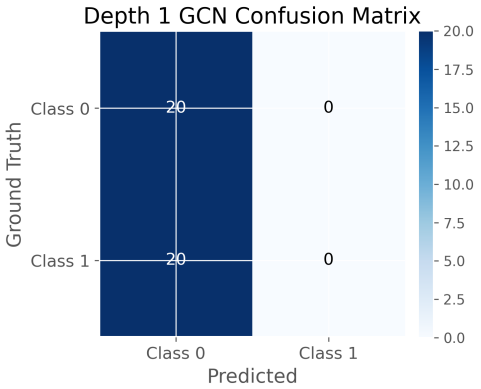
During training, we use Adam with a learning rate of 1e-3, a batch size of 32, and a maximum of 10 epochs. Early stopping is triggered when validation accuracy fails to improve for 3 consecutive epochs.

B ADDITIONAL FIGURES FROM DESIGN VARIANTS

We present additional confusion matrices and training curves for several model variants to highlight consistent pitfalls. Each approach (Depth-1 GCN, directed-edge GCN, fully connected GCN, sequence-order shuffled input, shape-only edges) exhibits similar overfitting behaviors, indicating that these issues are not confined to the baseline method.

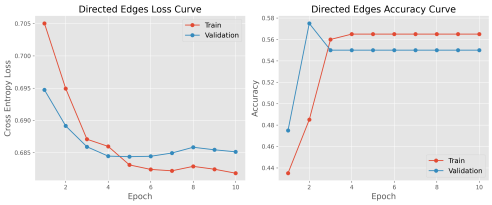


(a) Depth-1 GCN Loss/Accuracy

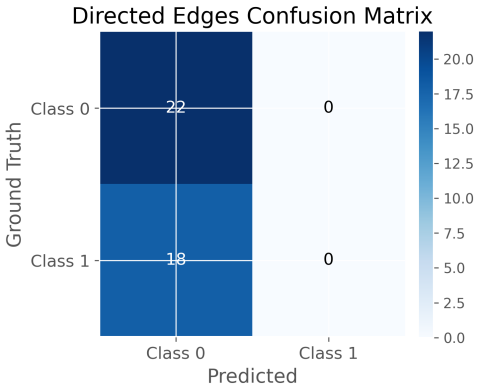


(b) Depth-1 GCN Confusion Matrix

Figure 2: Training curves and confusion matrix for Depth-1 GCN. Overfitting and a skew toward one class persist.

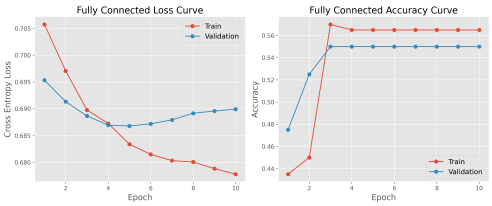


(a) Directed Edges Loss/Accuracy

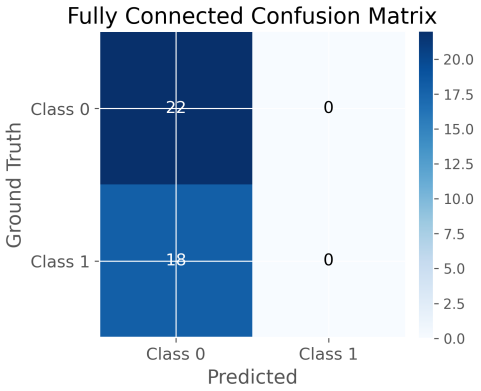


(b) Directed Edges Confusion Matrix

Figure 3: Directed-edge GCN approach, again showing similar pitfalls in training dynamics and prediction distribution.

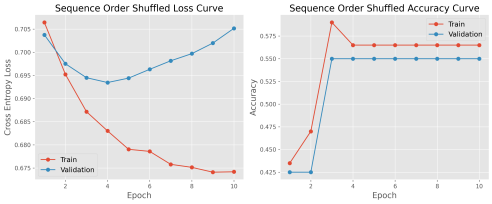


(a) Fully Connected Loss/Accuracy

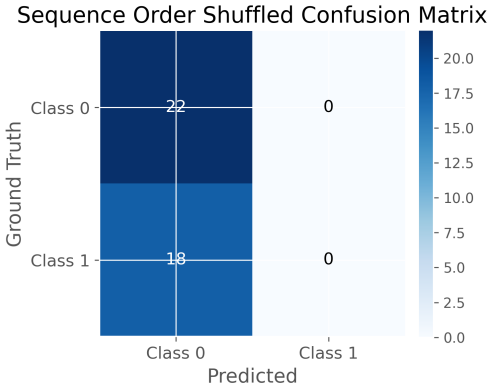


(b) Fully Connected Confusion Matrix

Figure 4: Fully connected GCN variant does not alleviate overfitting or class imbalance.

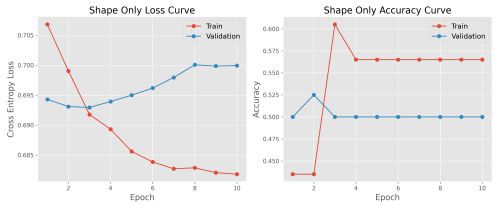


(a) Sequence Order Shuffled Loss/Accuracy

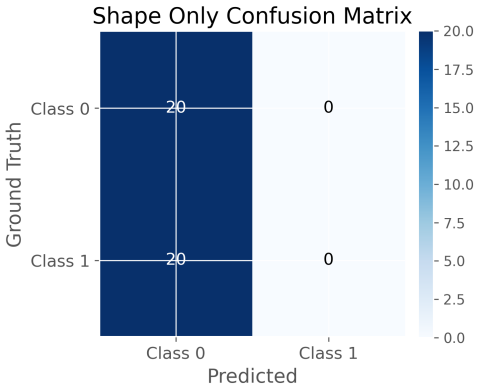


(b) Sequence Order Shuffled Confusion Matrix

Figure 5: Shuffled token order reveals similar overfitting, indicating that basic adjacency alone may not be the key factor.



(a) Shape-Only Loss/Accuracy



(b) Shape-Only Confusion Matrix

Figure 6: Shape-only edges degrade color-based reasoning but still reflect the same key pitfalls.