CONTEXTUAL EMBEDDINGS FOR COMPLEX SYMBOLIC RULE REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Synthetic PolyRule Reasoning (SPR) requires classifying sequences of abstract symbols under intricate rules. We explore whether contextual embeddings, widely used in NLP, can bolster accuracy on the challenging SPR_BENCH dataset. Despite adopting a Transformer-based model (Vaswani et al., 2017) adapted to symbolic input, our best test F1 score (79.8%) remains slightly below the 80.0% benchmark reported previously. We highlight partial successes, such as a discrete count-vector pathway that raises F1 from 79.5% to 79.8%. Persistent pitfalls underscore how difficult it can be to bridge linguistic embeddings and purely symbolic tasks, suggesting future work on specialized or hybrid approaches (Bortolotti et al., 2024; Lu et al., 2024) is warranted.

1 INTRODUCTION

Neural architectures excel at capturing contextual information in natural language processing (Vaswani et al., 2017; Ethayarajh, 2019), yet it remains unclear whether these embeddings can be repurposed for tasks outside language. One such task is Synthetic PolyRule Reasoning (SPR), which requires classifying symbolic sequences with complex, sometimes hidden rules. Success in this domain is vital for real-world settings that demand resilient pattern recognition (e.g., logistics pipelines, sensor data monitoring) under structured symbolic constraints.

Our objective is to adapt a Transformer architecture to SPR, examining whether it can surpass the previously reported 80.0% F1 on SPR_BENCH (Bortolotti et al., 2024). We highlight pitfalls encountered: expensive training, difficulties adapting embeddings to shape- or order-based patterns, and overfitting to spurious correlations. Our main contributions are: (1) an empirical analysis of Transformer-based contextual embeddings on an SPR setup, (2) a novel count-vector approach that slightly improves performance though it does not exceed the previous 80.0% mark, and (3) a discussion of real-world challenges in symbolic tasks, including the mismatch between linguistic embeddings and abstract feature constraints.

2 RELATED WORK

Symbolic reasoning often uses logic-based or hybrid neuro-symbolic systems (Lu et al., 2024), aiming to combine interpretability with deep-learning scalability. Transformers (Vaswani et al., 2017) have revolutionized NLP, capturing long dependencies efficiently, but their applicability to purely symbolic, order-based problems remains uncertain. Research in optimization (Goodfellow et al., 2016; Hwang, 2024; Pethick et al., 2025) has demonstrated that effective training is crucial for tasks with sensitive discrete structure. Our work advances this line of inquiry by directly testing contextual embeddings on SPR, emphasizing persistent gaps and pitfalls.

3 METHOD

We use a Transformer encoder operating on tokenized symbolic data. Each symbol is represented by a concatenation of character- and bigram-level embeddings, plus a sinusoidal positional encoding (Vaswani et al., 2017). A discrete count-vector pathway tallies symbol frequencies across the input and projects these counts through a two-layer MLP before concatenating with the Transformer

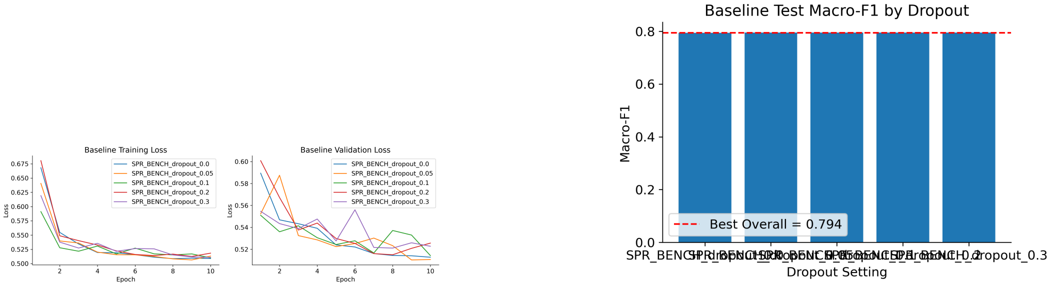


Figure 1: Effects of different dropout values on baseline Transformer. We show (Left) training/validation loss curves and (Right) test F1. Performance varies slightly but remains around 79.4%–79.5%.

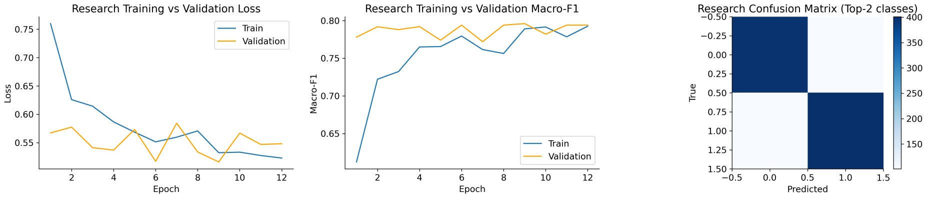


Figure 2: Extended model with count vector. (Left) Loss and Macro-F1 curves for training and validation exhibit volatility. (Right) Confusion matrix highlights persistent symbolic misclassifications.

encoding. We train using AdamW, applying gradient clipping for numerical stability (Goodfellow et al., 2016; Hwang, 2024; Pethick et al., 2025).

4 Experiments

We evaluated on SPR_BENCH (Bortolotti et al., 2024), splitting it into train/dev/test (50K/5K/5K). We initially tuned dropout (0.0 to 0.4) on the baseline Transformer. Figure 1 shows minor changes in training/validation loss and test F1 across dropout levels, with overall test F1 around 79.4%–79.5%. Adding our count pathway raises performance slightly to 79.8% but still falls short of 80.0%. Confusion matrices suggest persistent errors in shape-order classification and color-count matching, underscoring the difficulty of mapping aggregated linguistic embeddings to symbolic rules.

We extended the model with a count-vector pathway to handle discrete symbolic frequencies. As illustrated in Figure 2, training and validation metrics fluctuate over epochs, and the confusion matrix reveals systematic misclassifications for certain rule-based dependencies. While the overall F1 improves marginally, the results reinforce the need for stronger inductive biases or specialized structures.

5 Conclusion

Our exploration shows that contextual embeddings adapted from NLP are not a complete solution for symbolic reasoning tasks like SPR. Although adding a count-based pathway offers a small boost in F1, we remain below the 80.0% benchmark, indicating that linguistic embeddings struggle to capture symbolic constraints. From a practical standpoint, these pitfalls emphasize that direct application of NLP-based encoders can be misleading when underlying combinatorial or rule-based patterns require more specialized handling. Future directions could include incorporating stronger neuro-symbolic inductive biases (Lu et al., 2024), customized attention, or carefully curated data curricula to expose essential structures.

References

Samuele Bortolotti, Emanuele Marconato, Tommaso Carraro, Paolo Morettin, Emile van Krieken, Antonio Vergari, Stefano Teso, and Andrea Passerini. A neuro-symbolic benchmark suite for concept quality and reasoning shortcuts. 2024.

Kawin Ethayarajh. How contextual are contextualized word representations? comparing the geometry of bert, elmo, and gpt-2 embeddings. pp. 55–65, 2019.

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

Dongseong Hwang. Fadam: Adam is a natural gradient optimizer using diagonal empirical fisher information. *ArXiv*, abs/2405.12807, 2024.

Zhen Lu, Imran Afridi, Hong Jin Kang, Ivan Ruchkin, and Xi Zheng. Surveying neuro-symbolic approaches for reliable artificial intelligence of things. *J. Reliab. Intell. Environ.*, 10:257–279, 2024.

Thomas Pethick, Wanyun Xie, Mete Erdogan, Kimon Antonakopoulos, Tony Silveti-Falls, and V. Cevher. Generalized gradient norm clipping non-euclidean (l0,l1)-smoothness. *ArXiv*, abs/2506.01913, 2025.

Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and I. Polosukhin. Attention is all you need. pp. 5998–6008, 2017.

Supplementary Material

Here, we provide additional technical details, training configurations, and ablation results beyond what was shown in the main text.

Hyperparameters

We summarize our key hyperparameters:

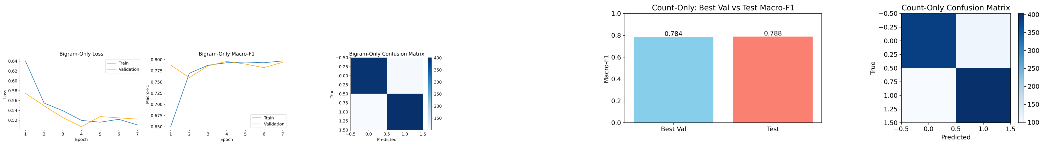
- **Transformer layers**: 4 layers, each with 8 attention heads.
- **Embedding size**: 128 for character-level; 128 for bigram-level.
- **Count-vector MLP**: 2 layers, 64 hidden units, ReLU activation.
- **Optimizer**: AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.999$, weight decay $= 10^{-5}$.
- **Gradient clipping**: norm of 1.0.
- **Batch size**: 64.
- **Learning rate**: 5e-4 after grid search.

Ablation Studies

We conducted ablation experiments to determine the importance of character-level embeddings and the utility of the count-vector pathway. Results confirm that neither removing character-level embeddings nor using purely count-based inputs outperforms the baseline, but each variant provides insight into symbolic vs. embedding-based features. Figure 3 merges two key ablations:

These ablations highlight that combining character embeddings with bigram tokens is beneficial. Meanwhile, relying solely on counts fails to capture rule-based patterns that often depend on ordering. We also tested disabling positional embeddings or removing Transformer context layers entirely; all such modifications degrade performance but do not provide additional insights beyond reinforcing that structured representations are necessary for SPR.

Code snippet. Below is an excerpt of our training pipeline, which loads SPR_BENCH, tokenizes inputs, and constructs either the baseline or count-vector-augmented model. Early stopping relies on dev Macro-F1:



(a) Bigram-only embeddings

(b) Count-only model

Figure 3: Ablations analyzing the impact of removing character-level embeddings (left) and removing token embeddings in favor of a purely count-vector approach (right). The bigram-only model slightly underperforms the full model, and the count-only approach struggles to model symbolic order.

```
def train_spr_model(args):
    dataset = load_spr_bench()
    model = MyTransformerModel(args)
    best_val_f1 = 0.0
    for epoch in range(args.max_epochs):
        train_epoch(model, dataset.train, args)
        val_f1 = evaluate(model, dataset.val)
        if val_f1 > best_val_f1:
            best_val_f1 = val_f1
            save_checkpoint(model)
```